INTERPRETING REINFORCEMENT LEARNING FOR POST-CAPTURE CONTROL IN SPACE DEBRIS REMOVAL

Jingyi Liu

INTERPRETING REINFORCEMENT LEARNING FOR POST-CAPTURE CONTROL IN SPACE DEBRIS REMOVAL

INTERPRETING REINFORCEMENT LEARNING FOR POST-CAPTURE CONTROL IN SPACE DEBRIS REMOVAL

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus, Prof. dr. ir. H. Bijl,
chair of the Board for Doctorates
to be defended publicly on
Thursday, 9 April 2026 at 12:30

by

Jingyi LIU

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	Chairperson
Prof. dr. E.K.A. Gill,	Delft University of Technology, *promotor*
Dr. J. Guo,	Delft University of Technology, *promotor*
Independent members:	
Prof. dr. G.C.H.E. de Croon	Delft University of Technology
Dr. ir. E. van Kampen	Delft University of Technology
Prof. dr. M.R. Lavagna	Politecnico di Milano, Italy
Dr. S. Theil	German Aerospace Center (DLR), Germany
Prof. dr. T. Keviczky	Delft University of Technology, reserve member

Keywords: Active Debris Removal, post-capture control, reinforcement learning, loss landscape, adaptive dynamic programming, attitude control, interpretation

Printed by: ProefschriftMaken

Cover by: Jingyi & Kai & AI

ISBN 978-94-6518-282-7

An electronic version of this dissertation is available at https://repository.tudelft.nl/.

A journey of research. A journey of self-discovery.

Contents

Summary

Space debris refers to defunct artificial objects in orbit and poses a major threat to on-orbit safety. Active Debris Removal (ADR) using robotic arms offers an effective solution to capture and remove debris. After debris capture, the resulting spacecraft can be regarded as a representative example of a system operating under uncertainty. Such uncertainty is not limited to post-capture scenarios but is a common feature of spacecraft attitude control, where parameter variations such as inertia uncertainty and external disturbances such as solar radiation pressure, affect system dynamics. Learning-based control approaches, particularly reinforcement learning (RL), provide a promising framework for dealing with such uncertain environments, as they can learn control behaviors without relying on explicit dynamic models. However, their performance varies across algorithms and applications, and the underlying learning behavior remains difficult to interpret. This research addresses this gap by developing visualization-based interpretation methods to analyze and explain the learning dynamics of actor–critic RL algorithms applied to spacecraft attitude control, thereby enhancing their interpretability and supporting their reliable application to space systems.

Reinforcement learning (RL) algorithms often show limited generalization across different dynamic systems. Such differences are closely related to how the critic network learns and optimizes its loss during training. To analyze this process, this work establishes a critic match loss landscape visualization method for online actor–critic algorithms. To capture the evolution of the critic during training, the network parameters are recorded at the end of each episode. To visualize their high-dimensional variation in a meaningful way, Principal Component Analysis (PCA) is applied to project the parameters onto a low-dimensional subspace. A fixed-target critic match loss is then defined using reference state samples and the corresponding temporal-difference (TD) targets associated with a selected policy, providing a consistent basis for comparison across training stages. The loss is evaluated over the principal component plane to generate a three-dimensional landscape and a two-dimensional contour with the training trajectory overlaid, thereby illustrating the critic's optimization behavior. To enhance the interpretability of the visualization, quantitative landscape indices are introduced to support systematic comparison across training outcomes. In addition, random-direction projections are employed to examine the consistency of the observed critic optimization patterns beyond a single PCA projection, reducing reliance on a specific projection direction. The Action-Dependent Heuristic Dynamic Programming (ADHDP) algorithm is employed to demonstrate the method on cart-pole and spacecraft attitude control tasks. The algorithm update process is explicitly revealed and characterized. The comparison of loss landscapes be-

tween the two systems shows how loss geometry corresponds to training stability and control performance, providing interpretable insights into critic optimization dynamics in online reinforcement learning control.

To broaden its applicability beyond online learning, the critic match loss landscape visualization method is extended to an off-policy reinforcement learning setting. The method is adapted to the Soft Actor–Critic (SAC) algorithm, which serves as a representative off-policy algorithm capable of producing convergent control results for the spacecraft attitude control problem. To accommodate the characteristics of SAC, the visualization procedure is refined to address its twin-critic structure, target computation, and replay-based training process. These adjustments align the method with the SAC learning mechanism while preserving its interpretive consistency with the online setting. Using this adapted method, critic match loss landscapes are reconstructed for SAC training and compared with those obtained for the online ADHDP algorithm. The resulting landscapes are analyzed both qualitatively and quantitatively using geometric indices together with temporal landscape snapshots. Comparative experiments reveal distinct loss geometries and optimization patterns across convergent and divergent learning cases, illustrating how different instability mechanisms correspond to characteristic critic optimization patterns that become visible through the joint analysis of loss geometry and optimization trajectories. These results demonstrate that the proposed visualization framework can interpret critic learning behavior and diagnose instability mechanisms in both online and off-policy actor–critic reinforcement learning algorithms.

Building on the previous method for visualizing critic learning behavior, this study extends it into a comprehensive framework for analyzing both actor and critic training processes. The framework is designed to provide a multi-perspective view of the learning dynamics, capturing how value estimation, policy optimization, and temporal-difference (TD) signals interact during training. It integrates four complementary visualization components: a three-dimensional critic match loss landscape that illustrates how the critic fits the TD targets and how the critic parameters evolve on the loss surface during training; an actor loss landscape that reveals the geometry and quality of the learned policy; a trajectory combining time, TD error, and actor weight evolution that depicts the coupling between policy updates and value estimation; and a state–TD plot that identifies state regions contributing to large TD fluctuations. Applied to multiple ADHDP variants with deep learning techniques for spacecraft attitude control, the framework reveals how these mechanisms reshape the optimization landscape and affect learning stability. Through this analysis, it enhances the interpretability of training behavior and offers insights that may inspire future algorithm design.

Together, these studies establish an interpretation framework for reinforcement learning in dynamic and control problems. The framework reveals how actor–critic algorithms learn and evolve in control tasks, providing an interpretable link between learning behavior and control performance. It deepens the understanding of internal learning mechanisms and offers diagnostic insights that can support further development and analysis of reinforcement learning control

methods. Future work may extend the framework toward dynamic visualization, theoretical analysis with stability considerations, and experimental validation through integrated physical platforms.

Samenvatting

Ruimteafval bestaat uit niet-functionerende kunstmatige objecten in een baan om de aarde en vormt een toenemende bedreiging voor de veiligheid van satellieten en ruimtevaartuigen. Actieve verwijdering van ruimteafval (Active Debris Removal, ADR) met behulp van robotarmen biedt een effectieve oplossing om dit afval op te vangen en te verwijderen. Na het vangen kan het resulterende ruimtevaartuig worden beschouwd als een representatief voorbeeld van een systeem dat onder onzekerheid opereert. Dergelijke onzekerheid is niet beperkt tot post-capture scenario's, maar is een algemeen kenmerk van de attitudecontrole van ruimtevaartuigen, waarbij parametervariaties en externe verstoringen de systeemdynamica beïnvloeden. Leergebaseerde controlemethoden, en in het bijzonder reinforcement learning (RL), bieden een veelbelovend kader om met dergelijke onzekerheid om te gaan, doordat zij controlegedrag kunnen aanleren zonder afhankelijk te zijn van expliciete dynamische modellen. De prestaties van RL-algoritmen verschillen echter tussen algoritmen en toepassingen, en het onderliggende leerproces blijft moeilijk te interpreteren. Dit onderzoek richt zich op dit vraagstuk door visualisatiegerichte interpretatiemethoden te ontwikkelen om de leerdynamiek van actor–critic-RL-algoritmen te analyseren en te verklaren, toegepast op de attitudecontrole van ruimtevaartuigen. Op deze manier wordt de interpreteerbaarheid vergroot en wordt een betrouwbaardere toepassing van RL in ruimtevaartsystemen ondersteund.

Reinforcement learning-algoritmen vertonen vaak beperkte generalisatie over verschillende dynamische systemen. Deze verschillen hangen nauw samen met de manier waarop het critic-netwerk leert en zijn verliesfunctie optimaliseert tijdens training. Om dit proces te analyseren wordt in dit werk een critic match loss landscape-visualisatiemethode ontwikkeld voor online actor–critic-algoritmen. Tijdens de training worden de parameters van het critic-netwerk aan het einde van elke episode opgeslagen. Om hun hoogdimensionale variatie op een betekenisvolle manier te visualiseren, wordt Principal Component Analysis (PCA) toegepast om de parameters te projecteren naar een laagdimensionale subruimte. Vervolgens wordt een vaste doelfunctie voor de critic match loss gedefinieerd op basis van referentietoestandmonsters en de bijbehorende temporal-difference (TD)-doelen die gekoppeld zijn aan een geselecteerd beleid, waardoor een consistente basis ontstaat voor vergelijking tussen verschillende trainingsfasen. Deze verliesfunctie wordt geëvalueerd in het PCA-vlak om een driedimensionaal landschap en een tweedimensionale contour met de trainingstrajecten te genereren, waarmee het optimalisatiegedrag van de critic zichtbaar wordt gemaakt. Om de interpreteerbaarheid van de visualisatie te vergroten, worden kwantitatieve landschapindices geïntroduceerd om systematische vergelijking tussen verschillende

trainingsresultaten te ondersteunen. Daarnaast worden projecties in willekeurige richtingen toegepast om de consistentie van de waargenomen optimalisatiepatronen van de critic te onderzoeken voorbij één enkele PCA-projectie, waardoor de afhankelijkheid van een specifieke projectierichting wordt verminderd. Het Action-Dependent Heuristic Dynamic Programming (ADHDP)-algoritme wordt gebruikt om de methode te demonstreren op zowel het kart-pendelsysteem als de attitudecontrole van een ruimtevaartuig. Het updateproces van het algoritme wordt hiermee expliciet zichtbaar en gekarakteriseerd. Vergelijking van de verlieslandschappen tussen beide systemen laat zien hoe de geometrie van het verlies samenhangt met trainingsstabiliteit en controleprestaties, en biedt interpreteerbare inzichten in de optimalisatiedynamiek van de critic in online reinforcement learning-besturing.

Om de toepasbaarheid van de methode verder uit te breiden voorbij online leren, is de critic match loss landscape-visualisatiemethode uitgebreid naar een off-policy reinforcement learning-omgeving. De methode is toegepast op het Soft Actor–Critic (SAC)-algoritme, dat fungeert als een representatief off-policy algoritme dat in staat is om geconvergeerde controleresultaten te bereiken voor het attitudecontrolesysteem van een ruimtevaartuig. Om rekening te houden met de specifieke eigenschappen van SAC, is de visualisatieprocedure verfijnd zodat deze de dubbele critic-structuur, de doelberekening (target computation) en het replay-gebaseerde trainingsproces kan verwerken. Deze aanpassingen stemmen de methode af op het leermechanisme van SAC, terwijl de interpretatieve consistentie met de online toepassing behouden blijft. Met deze aangepaste methode worden critic match loss landscapes gereconstrueerd voor SAC-training en vergeleken met die verkregen voor het online ADHDP-algoritme. De resulterende landschappen worden zowel kwalitatief als kwantitatief geanalyseerd met behulp van geometrische indices, in combinatie met temporale landscape-snapshots. Vergelijkende experimenten tonen duidelijke verschillen in verliesgeometrie en optimalisatiepatronen tussen geconvergeerde en divergerende leersituaties, en laten zien hoe verschillende instabiliteitsmechanismen overeenkomen met karakteristieke critic-optimalisatiepatronen die zichtbaar worden door de gezamenlijke analyse van verliesgeometrie en optimalisatietrajecten. Deze resultaten tonen aan dat het voorgestelde visualisatiekader het leergedrag van de critic kan interpreteren en instabiliteitsmechanismen kan diagnosticeren in zowel online als off-policy actor–critic reinforcement learning-algoritmen.

Voortbouwend op de eerdere visualisatiemethode wordt het concept verder uitgebreid tot een omvattend raamwerk voor de analyse van zowel actor- als critic-training. Het raamwerk is ontworpen om een meerperspectivisch beeld te bieden van de leerdynamiek, waarbij de interactie tussen waardeschatting, beleidsoptimalisatie en TD-signalen tijdens training wordt vastgelegd. Het raamwerk omvat vier complementaire visualisatiecomponenten:een driedimensionaal critic match loss-landschap dat laat zien hoe de critic de TD-doelen benadert en hoe de critic-parameters zich tijdens de training over het verlieslandschap ontwikkelen; een actor loss-landschap dat de geometrie en kwaliteit van het geleerde beleid weergeeft; een traject waarin tijd, TD-fout en gewichtsontwikkeling van de actor wor-

den gecombineerd om de koppeling tussen beleidsupdates en waardeschatting te illustreren; en een toestand–TD-plot die de toestandsgebieden identificeert die bijdragen aan grote TD-fluctuaties. Toegepast op verschillende ADHDP-varianten die deep reinforcement learning-technieken bevatten, laat het raamwerk zien hoe deze mechanismen het optimalisatielandschap hervormen en de leerstabiliteit beïnvloeden. Via deze analyse vergroot het raamwerk de interpreteerbaarheid van het leergedrag en biedt het kwalitatieve inzichten die toekomstige algoritmeontwerpen kunnen inspireren.

Gezamenlijk vormen deze studies een interpretatieraamwerk voor reinforcement learning in dynamische en controletoepassingen. Het raamwerk laat zien hoe actor–critic-algoritmen leren en zich ontwikkelen in controletaken en biedt een interpreteerbare koppeling tussen leergedrag en controleprestatie. Het verdiept het inzicht in interne leermechanismen en biedt diagnostische inzichten die de verdere ontwikkeling en analyse van reinforcement learning-controlemethoden kunnen ondersteunen. Toekomstig onderzoek kan het raamwerk verder uitbreiden richting dynamische visualisatie, theoretische analyse met stabiliteitsoverwegingen en experimentele validatie via geïntegreerde fysieke platforms.

Nomenclature

Abbreviations

Abbreviation	Meaning
AC	Actor-Critic
ADHDP	Adaptive Dual Heuristic Dynamic Programming
ADP	Adaptive Dynamic Programming
ADR	Active Debris Removal
DDPG	Deep Deterministic Policy Gradient
DP	Dynamic Programming
DPG	Deterministic Policy Gradient
ECI	Earth Centered Inertial
ESA	European Space Agency
IQR	Interquartile Range
JAXA	Japan Aerospace Exploration Agency
LEO	Low Earth Orbit
LEOP	Launch and Early Orbit Phase
MLP	Multi-Layer Perceptron
NN	Neural Network
PCA	Principal Component Analysis
PD	Proportional-Derivative
PID	Proportional–Integral–Derivative
PLE	Policy Linear Equation
PPC	Prescribed Performance Control
PPO	Proximal Policy Optimization
RELU	Rectified Linear Unit
RL	Reinforcement Learning
SAC	Soft Actor-Critic
TD	Temporal Difference
t-SNE	T-distributed Stochastic Neighbor Embedding
TPS	Target Policy Smoothing

Symbols and notations

Symbol	Description
ζ	Chosen center point in the landscape graph
$\boldsymbol{\delta}, \boldsymbol{\eta}$	Selected directions used to visualize the loss landscape

Symbol	Description
α, β	Distances from the center point along directions $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$
α^*, β^*	Selected perturbed parameters for calculation
$f(\alpha, \beta)$	Loss function evaluated at perturbed parameters
$\mathcal{L}$	Loss function operator used in the visualization
L^*	Critic match loss evaluated at the selected critic parameters
$L_{\text{match}}(\alpha, \beta)$	Critic match loss at perturbed parameters α, β
N_{snap}	Number of recorded critic weight snapshots used for PCA
$\mathbf{w}_c(k)$	Critic weight vector recorded at episode k
$\bar{\mathbf{w}}_c$	Mean critic weight vector of the recorded trajectory
$\mathbf{w}_c^r$	reference critic weight vector
$\tilde{\mathbf{w}}_c$	reconstructed critic weight vector
x_t^{state}	System state at time step t
u_t^{action}	Control action corresponding to state x_t^{state}
N_{ref}	Number of samples in the reference dataset
N_{snap}	Number of recorded critic weight snapshots
$\mathbf{X}$	Centered data matrix constructed from critic weight snapshots
$\mathbf{C}$	Sample covariance matrix of the centered critic weights
λ_i^{PCA}	Eigenvalue of the covariance matrix in PCA
$\mathbf{v}_i$	Eigenvector of the covariance matrix corresponding to λ_i^{PCA}
$\tilde{L}(\alpha, \beta)$	Normalized relative critic match loss
$\Delta L(\alpha, \beta)$	Relative critic match loss
ϵ	Radius of local neighborhood used for sharpness evaluation
θ_{dir}	Angular direction variable on the circle of radius ϵ
Sharp_ϵ	Sharpness index of the loss landscape
ρ	Loss threshold defining the low-loss basin region
A_ρ	Basin area where normalized loss remains below threshold ρ
H	Hessian matrix of the normalized loss on the PCA plane
$\lambda_{\max}, \lambda_{\min}$	Largest and smallest eigenvalues of Hessian matrix H
κ	Condition number of the Hessian matrix
$\log \kappa$	Logarithmic measure of local anisotropy of the loss landscape
$J(t)$	Cost function at time step t
$r(t)$	Reinforcement signal (reward) at time step t
r	Instantaneous reward
γ	Discount factor for future rewards
$c(x_t, u_t)$	Instantaneous stage cost used for performance evaluation
H_{eval}	Fixed evaluation horizon used for policy comparison
$J_{H_{\text{eval}}}$	Discounted cumulative cost over the evaluation horizon
$\tilde{J}_{H_{\text{eval}}}$	Normalized performance index
t_{fail}	Time step at which a failure event occurs
$c_{\max}$	Maximum normalized stage cost
$e_c(t), e_t^{\text{TD}}$	Temporal-difference error of the critic network
n	Dimension of the system state vector
$E_c(t)$	Instantaneous error function of the critic network
$E_a(t)$	Instantaneous error function of the actor network

Symbol	Description
$\mathbf{w}_c$	Weight parameters of critic network
$\mathbf{w}_a$	Weight parameters of actor network
s_t, a_t	System state and action at time step t
$\pi(a\|s)$	Policy mapping state s to action a
ρ_π	State–action distribution under policy π
d_π	State–action visitation distribution induced by policy π
$\mathbb{E}[\cdot]$	Expectation operator
$\mathcal{H}(\pi(\cdot\|s_t))$	Entropy of the policy distribution at state s_t
$J(\pi)$	Entropy-augmented policy objective in SAC
α_{temp}	Temperature coefficient in SAC
$\mathcal{D}$	Replay buffer or fixed probe dataset
$Q_{\mathbf{w}_c}(s_t, a_t)$	Q-value estimated by critic network
$\log \pi(a_{t+1}\|s_{t+1})$	Log-probability of sampled action
τ_{target}	Soft update coefficient for target critic
α_{cost}	Cost scaling factor used for numeric cost normalization
c_t	Instantaneous cost at time step t
c'_t	Scaled instantaneous cost used during training
$J(x)$	Value function associated with state x
$J'(x)$	Value function under the scaled cost
L_{critic}	Huber loss for critic network
e	Prediction error in Huber loss
κ_{huber}	Threshold parameter of the Huber loss
ℓ_1	Linear penalty term in the Huber loss
ℓ_2	Quadratic penalty term in the Huber loss
y_t^{td}	Target value for critic update at time step t
$\tilde{u}_{t+1}$	Noisy and clipped next action used in target policy smoothing
ϵ_{noise}	Policy smoothing noise added to the next action
σ_{noise}	Standard deviation of the policy smoothing noise
$\mathbf{I}$	Identity matrix defining independent noise in each action dimension
$u_{\max}$	Maximum limit for control torque
$\mathcal{L}_{\text{actor}}$	Actor loss evaluated over a fixed probe set
x	State vector in spacecraft attitude dynamics
u	Control input vector
P, Q	Weight matrices in quadratic reward function
$e_{\text{att}}, e_{\text{rate}}, e_{\text{torque}}$	Attitude, angular rate and control effort errors
$k_{\text{att}}, k_{\text{rate}}, k_{\text{torque}}$	Weighting coefficients for reward terms
$\mathcal{B}$	Body-fixed reference frame
$\mathcal{N}$	Earth Centered Inertial (ECI) reference frame
$\overrightarrow{\boldsymbol{n_1}}, \overrightarrow{\boldsymbol{n_2}}, \overrightarrow{\boldsymbol{n_3}}$	Unit vector of the $\mathcal{N}$ frame
$\vartheta_1, \vartheta_2, \vartheta_3$	Spacecraft attitude angles
$\omega_1, \omega_2, \omega_3$	Spacecraft angular velocities
$\bar{M}, \boldsymbol{M}$	Control torque applied to spacecraft attitude system
$\hat{J}_{\text{sc}}$	Spacecraft inertia matrix
$\boldsymbol{q} = [q_0, q_1, q_2, q_3]^T$	Unit quaternion representing spacecraft attitude

Symbol	Description
$\boldsymbol{\omega}$	Angular velocity vector of spacecraft
$\boldsymbol{\omega}_{\text{target}}$	Desired angular velocity vector
$\dot{\boldsymbol{q}}$	Time derivative of quaternion
$\otimes$	Quaternion multiplication operator
ψ	Angular displacement of the pole from upright position
$\dot{\psi}$	Angular velocity of the pole
$\ddot{\psi}$	Angular acceleration of the pole
h	Horizontal position of the cart
$\dot{h}$	Horizontal velocity of the cart
$\ddot{h}$	Horizontal acceleration of the cart
a	Normalized control action in range $[-1, 1]$
f	Force applied to the cart
F_{max}	Maximum allowable force magnitude
m	Mass of the pole
m_{c}	Mass of the cart
l	Half-length of the pole
g	Gravitational acceleration
μ_p, μ_{c}	Friction coefficients for the pendulum and the cart
$c(s, a)$	Instantaneous quadratic cost function
x_{max}	Maximum allowable cart displacement
$w_\psi, w_{\dot{\psi}}, w_h, w_{\dot{h}}, w_u$	Weighting coefficients in cart–pole cost function

1

INTRODUCTION

Space debris consists of inactive human-made objects and fragments in orbit, including defunct satellites, rocket bodies, and small mission-related items. This growing population has long been a concern, as it increases the risk of collisions with operational and servicing spacecraft. Active Debris Removal (ADR) has been proposed to actively remove debris from Earth orbit. Space debris is often an uncooperative target with unknown parameters or dynamics. As a consequence, in an ADR mission using robotic arms, the combined spacecraft after capture forms an uncertain system. Although this description comes from the ADR scenario, the resulting combined spacecraft represents a general class of attitude systems operating under uncertainty. Conventional control methods can be applied to post-capture attitude control of such combined spacecraft, and recent reinforcement learning algorithms also show promising performance for control under uncertainty. However, despite these results in dynamics and control, RL algorithms lack interpretability. Their performance and characteristics are difficult to explain. This research focuses on interpreting RL algorithms with application to post-capture attitude control of the combined spacecraft in ADR missions. This chapter reviews ADR missions, conventional and learning-based control methods, and existing approaches to RL interpretability. It then presents the background and motivation, followed by the research gaps and research questions. Finally, it outlines the dissertation and the research methodology for each chapter.

1

1.1. Background and motivation

Space debris comprises all inactive human-made objects and fragments in orbit, ranging from intact satellites and rocket bodies to small mission-related items [1]. The growth of the debris population has been a concern for decades since it adds to the collision possibilities for operational spacecraft. As a result, the concept of Active Debris Removal (ADR) has been proposed to remove space debris from the Earth's orbit actively.

Before debris can be removed, it must first be captured by an active spacecraft using robotic arms, tentacles, tethers, or other methods. In ADR missions, the debris to be captured is commonly referred to as the target object, which is typically non-cooperative and may tumble, fragment, or have unknown physical properties. Among the ADR capture methods, using robotic arms is a promising option, considering its validation and application in on-orbit servicing missions such as JAXA's ETS-VII [2]. In an ADR mission with robotic arms, some physical characteristics of the target could be observed or estimated before and during target capture [3]. Once the target is captured, the robotic arms are locked. During post-capture phase, the attitude stabilization of the combined spacecraft is achieved. The attitude stabilization control is a prerequisite for subsequent tasks, such as on-orbit servicing with robotic arms.

Although these descriptions arise from the ADR mission scenario, the combined spacecraft after capture can be viewed as a representative example of a combined spacecraft attitude system operating under uncertainty. Such uncertainty includes parameter variations, target-induced inertia changes, and external disturbances, and is not limited to ADR post-capture conditions. Similar forms of uncertainty commonly appear in spacecraft attitude control problems more generally.

The post-capture attitude control of the combined spacecraft can be achieved with conventional control methods, as well as reinforcement learning (RL) algorithms, which has been a research focus in recent years. These algorithms have demonstrated impressive performance in areas such as robotics, game playing, navigation, control, and decision-making. In ADR using robotic arms, the system to be controlled becomes a new combined spacecraft with uncertainties due to the uncooperative target capture, which is also a scenario where reinforcement learning algorithms can be applied [4].

However, despite the progress made in reinforcement learning control, its performance can vary across algorithms and applications. Since RL algorithms are commonly trained on a specific static environment, they may work well for one system yet fail to generalize to another [5]. Understanding why performance differs and how the learning process behaves is therefore important. Interpretation of reinforcement learning algorithms is useful for analyzing their internal mechanisms, identifying instability or poor generalization, and improving their reliability when applied to uncertain spacecraft dynamics.

This research focuses on interpreting reinforcement learning algorithms for a generalized combined spacecraft attitude control model, with the ADR post-capture phase serving as a representative scenario.

1.2. Literature review

This section provides a literature review on relevant topics about combined spacecraft control. It starts with the overview of the ADR mission, which briefly introduces the mission and analyzes the dynamic characteristics for the combined spacecraft in this specific scenario. Based on that, the dynamic system is generalized to a broader class of attitude systems with similar dynamic characteristics. It follows by the review of conventional control methods for attitude system with uncertainties. After that, learning-based control methods for control of uncertain system are reviewed . Finally, review on the interpretation of learning-based methods is given.

Active Debris Removal is increasingly recognised as a necessary complement to mitigation and design for removal measures as traffic in Low Earth Orbit (LEO) grows. Key capabilities have been demonstrated in flight. For example, the RemoveDEBRIS mission, led by the Surrey Space Centre, tested several in-orbit capture technologies, including net capture, a vision-based navigation experiment, and a harpoon. These tests provided early proof of concept for active debris removal techniques [6]. Astroscale's ELSA-d mission demonstrated controlled rendezvous and proximity operations with magnetic docking and repeated capture and release of a client spacecraft, which marked an important step for commercial end of life servicing [7]. In Europe, ClearSpace 1 is planned to deorbit the PROBA 1 satellite using a four armed capture mechanism [8]. ClearSpace 1 is regarded as the successor to ESA's e.Deorbit concept[9], which was initiated under the Clean Space activities. The stated objective of e.Deorbit was to remove a single large ESA owned debris object from LEO, and is often cited as the first active removal mission studied within ESA [9]. ENVISAT was used as a sizing case for a potential removal target [10]. Figure 1.1 provides an overview of the six use cases of this mission.

Targets are typically uncooperative, which means that the states and attributes of the targets are unknown. The dynamic characteristic of the combined spacecraft formed by space manipulator capturing uncooperative targets could be di-

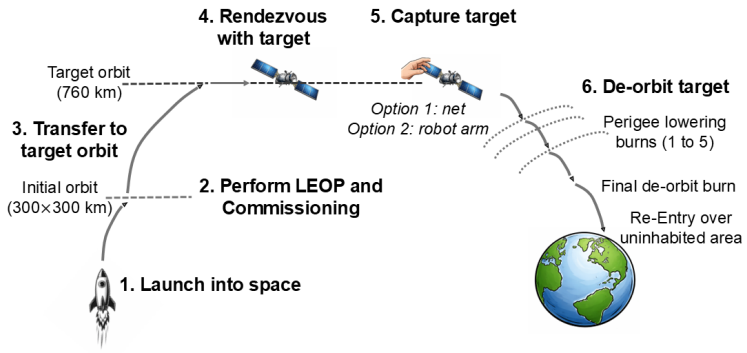


Figure 1.1: e. Deorbit use cases [9]

1

vided into the following two categories based on whether there is docking interface between the target and the operational spaecraft[11], as shown in Table 1.1.

Table 1.1: Uncooperative levels of targets [11]

Docking interface	**Characteristics**	**Examples**
Existent	Dockable & unpredictable	Inactive satellites
Non-existent	Non-dockable	Fragmentation

For the first category, the newly formed combined spacecraft could ideally be treated as a single rigid spacecraft with unknown inertia parameters. When there is docking interface between the space manipulator and the target, the latching end-effector can ensure a rigid connection between the target and end-effector. Once the target is captured, the robotic arms are locked. Then there is no relative motion between the target and the operational spacecraft or time-varying inertia parameters in the combined spacecraft. Then the combined spacecraft can be regarded as a single rigid spacecraft with unknown inertia parameters if both bodies have no internal slosh or flexibility.

For the second category, the new formed combined spacecraft could be treated as a multibody spacecraft system with unknown time-varying inertia.Since there is no docking interface between the space manipulator and the target, the end-effector configuration does not achieve force closure. Thus, possible relative sliding motion may occur between the target and end-effector if the friction coefficient is low [2]. Moreover, if the target itself is a multibody system, the resulting combined spacecraft will become an even more complex system with highly uncertain dynamics, which significantly complicates controller design.

The dynamic characteristics of the combined spacecraft in the ADR mission have been analyzed. Based on the level of system uncertainty, the problem is further generalized into a broader class of uncertain systems. As shown in Figure 1.2, the figure summarizes the overall structure of this literature review, including sections organized by uncertainty levels as well as learning-based control and interpretation.

1.2.1. Conventional attitude control methods of combined spacecraft

In this part, literature review on using conventional control methods for attitude control of combined spacecraft is given. It starts with the lowest level of system uncertainty, which is the parameter uncertainty. And conventional control methods for system with uncertain dynamics are reviewed after that.

Parameter-based control methods under unknown parameters

In the ADR scenario, after the target is captured, the service spacecraft changes its configuration and dynamic parameters, such as the shifted center of mass, due to the influence of the target spacecraft. Effectively dealing with the impact

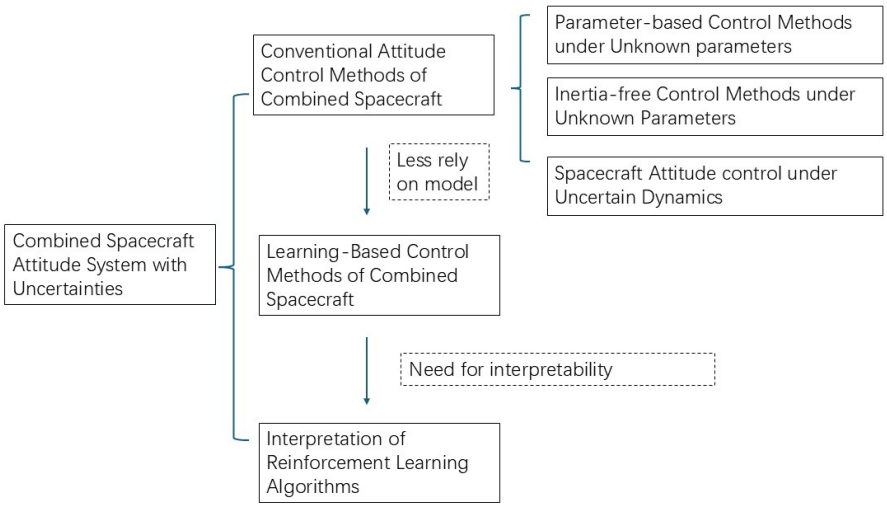


Figure 1.2: Structure of literature review

of target capture is the key to controlling the combined spacecraft. There are two main types of common control methods: 1) Treat the impact of the target on the combination as disturbance, and directly design a robust adaptive attitude control algorithm. 2) First identify the target quality characteristics, and then design the controller based on the identification results, which is similar to attitude stabilization control with cooperative targets yet taking parameter uncertainties into consideration.

For takeover control, representative studies include adaptive fault tolerant laws under actuator faults [12, 13], state dependent Riccati equation based optimal and sub optimal control [14], robust nonlinear tracking with bounded disturbances and input to state stability properties [15], sliding mode designs such as dual integral schemes with reaction wheel reconfiguration and nested inner and outer loops [16, 17], modelling with extra degrees of freedom at the contact and first order sliding mode control to reduce the associated velocities [12], strategies for actuator failure and saturation including using the manipulator to provide additional torque [18], and command filtering adaptive backstepping for targets with partial failure [19]. These control methods form the main control methods for takeover control.

Inertia-free control methods under unknown parameters

Although controller design based on explicit parameter identification can be used to address unknown inertial parameters when capturing uncooperative targets, such approaches rely on estimating the system inertia through identification or estimation algorithms. In practice, the accuracy of the estimated parameters may be affected by modeling errors, measurement noise, and structural un-

certainties, particularly in capture and post-capture scenarios where the inertia properties may change significantly. Consequently, control methods that do not rely explicitly on knowledge of the inertial parameters have attracted increasing attention in recent years, which will be discussed in this subsection.

Adaptive methods have been widely used to address parametric uncertainty, where adaptive laws are designed to update controller parameters online to compensate for unknown parameters [20]. In contrast to explicit parameter identification approaches, adaptive control integrates parameter estimation into the control law itself and adjusts the controller parameters directly according to system responses. These considerations motivate a closer look at how adaptive methods have been applied in combined spacecraft attitude stabilization.

Considering the scenario of capture and post-capture combined spacecraft attitude stabilization, a number of studies have been carried out in recent years. In [21], a nonlinear direct adaptive output control method was proposed to address the large inertia uncertainty introduced by the space station manipulator arm during target capture. The control law was derived from the simple adaptive control framework. In [19], taking system uncertainty into account, an improved dynamic inversion control law was designed with the aim of enabling thrust reallocation. In [22], a two-step control scheme for the attitude control of a multibody spacecraft system was developed. First, the actual system was estimated and a nominal controller was designed based on the estimated model. Then an additional term was included to compensate for the deviation between the actual and estimated systems. This controller can handle the multi-body attitude control problem with unknown parameters, but it requires known bounds on the parameter estimates. In [23], a robust inertia-free attitude takeover control method for the post-capture configuration was proposed, considering unmeasured states, unknown inertia characteristics, and external disturbances. This method does not rely on parameter identification, neural networks, or fuzzy approximation, which simplifies the control design.

If the relative motion between the service spacecraft and target is neglected, the combined spacecraft could be regarded as a single rigid spacecraft with unknown parameters. The inertia-free attitude stabilization control for single rigid spacecraft will be discussed. In [24], an indirect robust fault-tolerant controller that could simultaneously handle external interference and the unknown time-varying moment of inertia of the spacecraft was proposed, which would not demand substantial on-board computing resources. In [25]and [26], a class of adaptive nonlinear attitude controllers was designed. These controllers also prevented the unwinding phenomenon, a well-known issue in quaternion-based control where the controller unnecessarily rotates the spacecraft through a long path despite a shorter rotation being available. In [27] and [28], a continuous feedback controller was designed for a large number of spacecrafts, considering the unknown rotational inertia, and its approximate global stability characteristics were analysed. In [29], robust output feedback attitude stabilisation was studied for an uncertain spacecraft system subject to external disturbances, parameter uncertainty, unknown inertia, gain perturbations, measurement errors, and input

saturation. The design ensures the $\mathcal{H}_\infty$ performance, i.e., the worst-case gain from the lumped disturbance and measurement error to the output, remains below a prescribed attenuation level while maintaining bounded control input. Existence conditions are derived as linear matrix inequalities using Lyapunov theory.

Spacecraft attitude control under uncertain dynamics

For spacecraft attitude control under uncertain dynamics, it can be catogorized into model-based or model-free control methods. Many model-based control methods have been proposed to improve the closed-loop performance, such as slide mode control methods [30], finite-time control methods [31], backstepping control methods [32], and adaptive control methods [20]. These methods indeed address the attitude control problem under modelling errors, unmodeled dynamics, and external disturbances.

These model-based control methods usually require knowledge of the models and parameters of spacecraft, which makes a closed-loop system to be sensitive to model errors or parameter variations. Considering the post-capture attitude control for combined spacecraft, the dynamic models for traditional rigid spacecraft or spacecraft with flexible appendages can be established mathematically. However, it is difficult to establish a mathematical model for the combined spacecraft which contain relative motions. For example, if the target does not have a docking interface, as shown in Table 1.1, it's hard to derive the precise dynamic model for the combined spacecraft. Therefore, it is necessary to develop a model-free method.

The traditional proportional–integral–derivative (PID) control method exhibits a classical idea for model-free control, which is a linear combination of the present, accumulative and predictive forms of tracking error to construct the control input. Because of its simplicity, the PID control method is widely applied in many fields, including industrial process control, robotics, manufacturing automation, and flight control systems. However, conventional PID control may suffer from limited performance when dealing with complex nonlinear systems, time-varying dynamics, or significant uncertainties. To improve the reliability of the PID method, a variety of intelligent PID methods have been proposed, such as fuzzy PID control and heuristic PID control [33, 34].

Prescribed performance control (PPC) introduces a model-free perspective [35]. It specifies the desired transient and steady-state behaviour by imposing performance functions on system states or errors. In the attitude control of combined spacecraft, PPC can be effective even without an explicit dynamics model or disturbance information, and it can be applied to both rigid and flexible systems [36]. The specific forms of PPC vary depending on the controller design, and adaptive variants often employ neural networks, fuzzy systems, or support vector machines to approximate unknown dynamics online [37–40]. Despite their strong performance, such identification and adaptation mechanisms may require substantial computational effort.

With the continuous development of new technologies, especially information technology and computer technology, a large amount of data has been accumu-

lated in industrial production processes. This provides the necessary preconditions for the emergence and development of data-driven or data-dependent control. In simple terms, data-driven control refers to controller design that does not require an explicit system model but relies solely on online or offline input–output data of the plant, while still ensuring convergence and stability [41].

Several studies have applied data-based methods to attitude stabilization control. Typical data-based approaches include policy iteration and PLE-based schemes, where PLE (Policy Linear Equation) methods estimate the value function or control gain by solving linear equations constructed directly from data rather than from a known model. Jiang et al. [42] presented a computational policy-iteration approach for continuous-time linear systems with unknown dynamics. Their method solves the algebraic Riccati equation online using state and input data, and a practical implementation was demonstrated on a turbocharged diesel engine. Jiang et al. [43] proposed an iterative PLE method with an initial stabilizing gain for attitude systems, proved closed-loop stability and convergence, and developed a data-driven controller using only input and state measurements.

Conventional control methods have long been used for spacecraft attitude systems with uncertainty and can provide reliable performance with clear stability guarantees. However, their effectiveness often depends on having accurate models, known bounds on uncertainties, or parameters that can be well characterized. In practical situations, these conditions are not always easy to meet, especially when the spacecraft operates in complex or changing environments. When the system becomes strongly nonlinear, contains large or rapidly varying uncertainties, or involves unknown interactions such as those in the capture and post-capture combined spacecraft dynamics, model-based designs may be difficult to apply or may lead to conservative controllers. These challenges motivate the use of learning-based control methods, which aim to make use of data to handle complex or partially known dynamics and offer a more flexible option in such scenarios. Learning-based approaches may also reduce the reliance on detailed modelling and allow the controller to adjust its behaviour as new information becomes available.

1.2.2. Learning-based control methods of combined spacecraft

Learning-based control methods have been increasingly explored as an alternative to conventional control methods designs. These approaches make use of data to improve control performance when system models are incomplete or difficult to obtain. In spacecraft attitude control, various learning-based methods have been developed to address complex dynamics and uncertainties.

Learning-based methods have been applied to attitude stabilization of combined spacecraft with unknown dynamics. In [44], a terminal sliding mode controller with a Chebyshev neural network was proposed to achieve finite time stabilization under unknown inertia. In [45], a learning-based adaptive takeover controller was developed for unknown inertia and external disturbances: a static prescribed performance controller guarantees uniformly ultimately bounded tracking errors, and an adaptive dynamic programming based supplement improves

robustness and adaptation using only input and output data, without inertia identification. In [46], a quasi model-free scheme was presented that builds an online excitation response mapping database and introduces a virtual coordinate system to compute the time varying centroid. A linear online corrector reduces tracking errors and enables high precision control under adverse conditions.

Reinforcement learning is a promising way to deal with “unknowns.” In machine learning, common categories include supervised learning, unsupervised learning, and reinforcement learning. Supervised and unsupervised learning rely on fixed datasets, whereas reinforcement learning modifies actions through interaction with the environment. In RL, an actor or agent interacts with its environment and adjusts its actions, or control policies, based on the stimulus or feedback received in response to those actions [47]. The central idea is that successful control decisions should be reinforced so that they become more likely to be used again. Although this idea originates from experimental animal learning, RL is theoretically closely related to direct and indirect adaptive optimal control methods.

Dynamic programming (DP) is a classical and powerful tool for solving optimal control problems. Nevertheless, DP is typically implemented offline and becomes increasingly difficult to apply when the system dimension grows, due to the well-known curse of dimensionality [48]. To enable the usage of DP forward in time, the adaptive dynamic programming (ADP) framework was developed. ADP combines reinforcement learning (RL) ideas with the actor–critic (AC) structure [49] and has therefore received substantial attention over the past decades. In this framework, an additional excitation signal is introduced so that the controller can interact with the unknown environment. Using the available measurable data, the cost function is updated and optimized, producing control actions that approximate the optimal solution and achieve the desired system performance. Due to its advantages in solving optimization problems, the ADP method has received extensive attention in the past ten years, such as the water gas transition reaction control [50], power system control, etc. [51]. In the learning algorithm of ADP, two methods of policy iteration and value iteration are usually used to realize the optimization process. For policy iteration, the ADP learning process must start with a stable initial allowable control policy, otherwise the learning process will diverge. For value iteration, although the shortcomings of policy iteration are overcome, the stability of the value iteration process is difficult to ensure stability during the entire learning process [52, 53].

For on-orbit tasks such as post-capture attitude stabilization of a combined spacecraft or coordinated control of spacecraft formations, high-quality control typically relies on continuous interaction and feedback between the system and its environment. This closed-loop process is conceptually similar to the learning mechanism in ADP. Therefore, ADP methods have attracted interest and have been explored in previous studies. However, their application to such systems still requires careful consideration due to the nonlinear dynamics, uncertainties, and strict stability requirements of spacecraft control.

The actor-critic architecture, sometimes referred to as the adaptive-critic frame-

work, is widely employed to implement generalized policy iteration in reinforcement learning [54]. In this approach, the actor network is used for generating control policies, while the critic network evaluates the expected cost-to-go. Actor networks can be trained in different ways, such as directly minimizing the estimated cost-to-go provided by the critic or by reducing the temporal-difference (Bellman) error, which measures the difference between the current estimated value of critic and the value predicted by the Bellman equation. The critic, in turn, can be updated using temporal difference methods or gradient-based techniques [55]. Together, the actor and critic work iteratively to improve policy performance and approximate optimal decision-making for discrete-time Markov decision processes.

Actor critic methods such as deterministic policy gradients[56], twin delayed variants[57], soft actor critic[58], and proximal policy optimisation[59] are commonly used because they handle continuous actions and allow direct tuning of performance objectives. Learning-based controllers have been reported to track large angle slews, tolerate unknown or varying inertia, and operate with limited model information. Hybrid designs are frequent, where reinforcement learning augments a baseline controller, an extended state observer, or an adaptive element to improve disturbance rejection and reduce modelling effort[60].

Methodological trends include the use of curriculum learning, domain randomisation, and demonstration data to improve sample efficiency and robustness, together with explicit constraint handling through barrier functions, safety filters, or integration with model predictive control. Attention has also turned to diagnostic and interpretive tools that relate value function quality, temporal difference behaviour, and weight space evolution to closed loop performance, which is important for high risk missions [61]. Open challenges remain in safety assurance, data efficiency under realistic sensor and actuator limits, and reliability when dynamics change during operations such as post capture assembly. These concerns motivate research on reproducible evaluation, principled interpretation, and design guidance that connects learning signals to controller choices in space applications.

For on-orbit reconfiguration, where a spacecraft adjusts its configuration or operational state to meet new mission requirements under multiple constraints, a dual-objective adaptive dynamic programming scheme trains mission and energy networks with Q-learning to approximate the Hamilton–Jacobi–Bellman solution and balance mission reward with control cost [62]. Output-feedback designs combine an extended state observer with a proportional–derivative baseline and an ADHDP (Action-Dependent Heuristic Dynamic Programming) supplement, where ADHDP uses both actor and critic networks to evaluate and improve control actions, to enhance tracking under uncertainties [63]. Finite-horizon optimal control for linear time-varying discrete-time systems has been addressed with data-driven policy iteration and value iteration that learn optimal controllers from input and state data without a model [64]. Switched attitude control with four thrusters formulates the system dynamics as a set of subsystems, uses dynamic programming to select the optimal switching strategy, and employs a

neural-network-based cost approximation trained offline [65]. Deep reinforcement learning, a class of RL methods that employ deep neural networks to approximate policies or value functions, has produced adaptive controllers that achieve large-angle slews, transfer across spacecraft, and tolerate disturbances unseen in training [66], while proximal policy optimisation with curriculum learning improves robustness under unknown inertia [67]. Incremental approximate dynamic programming combined with nonlinear control has been used to realise adaptive nonlinear tracking while reducing the rapid growth of computational effort that arises as state dimension increases [68]. In addition, deep reinforcement learning has been applied to six-degree-of-freedom powered descent and landing with proximal policy optimisation, yielding accurate and fuel-efficient trajectories [69], and to proximity-operations guidance where feedback policies are learned for constrained relative motion [70].

Learning-based control methods provide a way to handle cases where the system dynamics are complicated or not fully known, and they have shown good potential in spacecraft attitude control. At the same time, their behaviour can depend on training data, network structure, and the internal learning process, making it difficult to understand why a controller performs well in some situations but not in others. To better analyse these methods, it is useful to have tools that can show how reinforcement learning algorithms update during training. This motivates the study of RL interpretation in the next section.

1.2.3. Interpretation of reinforcement learning algorithms

Among the learning-based methods in subsection 1.2.2, reinforcement learning learning has become a key approach for continuous control in robotics and aerospace, due to its ability to learn policies directly from interactions with the environment. Nevertheless, RL policies are often fragile and difficult to interpret agents trained in narrowly defined settings may fail to generalize when dynamics, noise, or other conditions change, a situation frequently encountered in real-world control tasks. Empirical studies have demonstrated that even small variations in environmental conditions or system parameters can lead to significant performance degradation, emphasizing the need to understand not only the behavior of RL policies, but also the mechanisms through which they learn and the reasons they fail under uncertainty [5, 71] .

Interpretation for RL in control can be categorized into three categories. The first and most active in continuous control is visualization-based interpretation, which examines the geometry of optimization and the trajectories taken during training. Drawing from deep-learning work on loss-landscape geometry, [72] introduced filter-normalized visualizations that relate minima sharpness and curvature to generalization. This idea has catalyzed analogous tools for RL. In RL specifically, reward and return landscapes expose how small policy perturbations map to expected return, revealing “cliffs” and noisy neighborhoods that correlate with instability and divergence during training [73, 74]. These views help explain why two runs of the same algorithm can exhibit markedly different outcomes and how stepping directions can move a policy into fragile regions of parameter

space. A complementary line analyzes how action representations reshape the underlying optimization surface and, in turn, learning dynamics in policy-gradient methods. The relationship between representation choices and observable differences in landscape complexity and convergence behavior has been investigated in [75]. Together, these visualization studies agree with the observation that the “loss landscape" or "return landscape" is a useful index for peering into internal training processes and diagnosing sensitivity in actor–critic control.

The second category focuses on policy extraction and simplification to yield globally interpretable, verifiable controllers. Decision-tree policies compress high-performing neural policies into structured trees that can be analyzed and formally checked, offering an interpretable surrogate for deployment or auditing [76]. Such methods clarify the underlying decision structure and support safety reasoning, but they may not scale well to high-dimensional continuous observations and can inherit limitations from the original neural policy.

The third category uses explanatory and attribution techniques to provide local, human-readable accounts of behaviour. Natural-language rationales can increase transparency of robot controllers by describing policy choices in context [77]. Saliency-style visual explanations highlight which parts of the observation influence actions and how attention shifts over learning [78]. These methods help users detect spurious correlations or over-reliance on specific cues, though they are typically post-hoc and do not directly expose learning dynamics or stability properties.

In summary, visualization-based interpretation methods allow a straightforward understanding of the RL algorithms in control. The interpretation for RL control has evolved from treating policies as black boxes to developing tools that connect optimization geometry with robustness. There are two problems that needs attention. Generalization beyond the training environment is fragile, so interpretation should focus on pre-mission verification and validation; Optimization geometry which is reflected in the structure of loss and return surfaces, as well as in simple projections, is a helpful sign to show stability or instability. For safety critical control applications, including robotics and spacecraft attitude control, a practical approach is to combine visualization tools that reveal training dynamics, e.g., loss landscapes and parameter trajectories, with both global and local explanations, thereby generating insights into policy stability, sensitivity, and potential failure modes under uncertain dynamics.

1.3. Research gaps and research questions

From the literature review, three research gaps can be identified.

Gap 1: Lack of interpretability methods tailored to RL in dynamic and control settings

RL algorithms, particularly those applied to continuous control systems, remain largely opaque. Existing interpretability methods are either developed for classification tasks or offer limited insight into the training behaviour of

actor–critic structures under dynamic conditions. There is a need for tools that can reveal how learning unfolds during training.

Gap 2: Absence of a generalizable framework for interpreting RL algorithm behaviour

While some interpretability techniques have been proposed, they tend to be algorithm-specific or lack a unified structure for cross-comparison. There is currently no standardized framework that enables the systematic visualization and comparison of RL learning dynamics across different algorithm variants and training setups.

Gap 3: Limited application of interpretability methods in spacecraft attitude control

In space applications such as post-capture attitude control, RL methods are increasingly used to handle nonlinear dynamics and real-time adaptation. However, existing work primarily focuses on performance metrics, with limited effort toward interpreting the learning process itself. Few studies have explored how interpretability tools can provide insight into the learning behaviour of RL algorithms in this domain. Furthermore, most existing approaches emphasize system-level behaviour, rather than algorithm-level learning characteristics, which are the focus of this thesis.

Based on the literature review and the identified research gaps, the following research questions are formulated.

RQ1. How can the performance of RL control algorithms be interpreted in systems with uncertainty?

a. How can an interpretation method be established to analyze the performance of online RL algorithms? (Chapter 2)

b. How can the interpretation method be applied to off-policy RL algorithms? (Chapter 3)

RQ2. How can the method identified in RQ1 be extended into a practical framework and applied to spacecraft attitude control? (Chapter 4)

a. Building on RQ1, how can the interpretation method be extended into a cohesive visualization framework?

b. How can this framework explain performance differences among ADHDP algorithm variants?

1.4. Methodology and thesis outline

The thesis follows a logical progression consisting of three stages: method establishment, method adaptation, and framework generalization. The research

questions are addressed across three main chapters. Chapter 2 addresses Research Question 1.a by establishing an interpretation method for online reinforcement learning control. Chapter 3 completes Research Question 1 by adapting the method to an off-policy reinforcement learning algorithm, thereby addressing Research Question 1.b. Chapter 4 addresses Research Question 2 by extending the method into a reproducible interpretation framework and applying it to explain performance differences among variants of an online reinforcement learning controller for spacecraft attitude control.

Chapter 1 introduces the research background, motivation, and questions, outlining the need for interpretable reinforcement learning control in uncertain spacecraft attitude systems. Chapter 5 concludes the thesis by summarizing the main contributions and suggesting directions for further work.

Chapter 2 establishes the visualization-based interpretation method that answers Research Question 1.a. A critic match loss landscape approach is developed to analyze the learning dynamics of actor–critic algorithms during online control. The method reveals how the critic's optimization behavior corresponds to learning stability and control performance, using the Action-Dependent Heuristic Dynamic Programming (ADHDP) algorithm as a representative example. Chapter 3 extends the method to address Research Question 1b by adapting it to an off-policy reinforcement learning context. The Soft Actor–Critic (SAC) algorithm is adopted as a convergent off-policy framework for spacecraft attitude control. This chapter demonstrates how the method can be aligned with different learning mechanisms, showing its interpretive consistency across online and off-policy actor–critic paradigms. Chapter 4 addresses Research Question 2 by expanding the method into a comprehensive interpretation framework. The framework integrates multiple complementary visualization perspectives to jointly analyze actor and critic training behavior. Applied to several ADHDP variants incorporating deep reinforcement learning techniques such as loss shaping, target updates, and training stabilizers, it qualitatively reveals how these mechanisms influence optimization geometry and learning stability. The framework thereby enhances the interpretability of reinforcement learning training processes and provides qualitative insights that may inform future algorithm design.

Together, these chapters form a coherent methodology, which first establishes the interpretation method, then extends it across learning settings, and finally integrates it into a unified framework for interpreting reinforcement learning in dynamic and control problems.

References

[1] *ESA's Annual Space Environment Report*. Tech. rep. GEN-DB-LOG-002. Definition of space debris and status of the space environment. European Space Agency, Space Debris Office, 2024. url: https://www.sdo.esoc.esa.int/environment_report/.

[2] M. Oda. "Space robot experiments on NASDA's ETS-VII satellite-preliminary overview of the experiment results". In: *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*. Vol. 2. IEEE. 1999, pp. 1390–1395.

[3] A. Flores-Abad, O. Ma, K. Pham, and S. Ulrich. "A review of space robotics technologies for on-orbit servicing". In: *Progress in aerospace sciences* 68 (2014), pp. 1–26.

[4] R. Cui, C. Yang, Y. Li, and S. Sharma. "Adaptive Neural Network Control of AUVs With Control Input Nonlinearities Using Reinforcement Learning". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 47.6 (2017), pp. 1019–1029. doi: 10.1109/TSMC.2016.2645699.

[5] C. Packer, K. Gao, J. Kos, P. Krähenbühl, V. Koltun, and D. Song. "Assessing generalization in deep reinforcement learning". In: *arXiv preprint* (2018). arXiv:1810.12282.

[6] Forshaw. *Dynamics and Relativity*. John Wiley & Sons, Mar. 5, 2009. 338 pp. isbn: 0470014601. url: https://www.ebook.de/de/product/8167643/forshaw_dynamics_and_relativity.html.

[7] C. Blackerby, A. Okamoto, S. Iizuka, Y. Kobayashi, K. Fujimoto, Y. Seto, S. Fujita, T. Iwai, N. Okada, J. Forshaw, *et al.* "The ELSA-d end-of-life debris removal mission: preparing for launch". In: *Proceedings of the International Astronautical Congress, IAC*. Vol. 8. 2019.

[8] R. Biesbroek, S. Aziz, A. Wolahan, S. Cipolla, M. Richard-Noca, and L. Piguet. "The clearspace-1 mission: ESA and clearspace team up to remove debris". In: *Proc. 8th Eur. Conf. Sp. Debris*. 2021, pp. 1–3.

[9] R. Biesbroek, L. Innocenti, A. Wolahan, and S. M. Serrano. "e. Deorbit-ESA's active debris removal mission". In: *Proceedings of the 7th European conference on space debris*. Vol. 10. ESA Space Debris Office Darmstadt, Germany. 2017.

[10] M. Wieser, H. Richard, G. Hausmann, J.-C. Meyer, S. Jaekel, M. Lavagna, and R. Biesbroek. "E. Deorbit mission: OHB debris removal concepts". In: (2015).

[11] M. Shan, J. Guo, and E. Gill. "Review and comparison of active space debris capturing and removal methods". In: *Progress in aerospace sciences* 80 (2016), pp. 18–32.

[12] Z. Wang, J. Yuan, and D. Che. "Adaptive attitude takeover control for space non-cooperative targets with stochastic actuator faults". In: *Optik* 137 (2017), pp. 279–290.

[13] Y. Xiaokui, T. Zhang, D. Honghua, N. Xin, and Y. Jianping. “Postcapture stabilization of space robots considering actuator failures with bounded torques”. In: *Chinese Journal of Aeronautics* 31.10 (2018), pp. 2034–2048.

[14] P. Huang, M. Wang, Z. Meng, F. Zhang, and Z. Liu. “Attitude takeover control for post-capture of target spacecraft using space robot”. In: *Aerospace Science and Technology* 51 (2016), pp. 171–180.

[15] S. Bandyopadhyay, S.-J. Chung, and F. Y. Hadaegh. “Nonlinear attitude control of spacecraft with a large captured object”. In: *Journal of Guidance, Control, and Dynamics* 39.4 (2016), pp. 754–769.

[16] D. Han, P. Huang, X. Liu, and Y. Yang. “Combined spacecraft stabilization control after multiple impacts during the capture of a tumbling target by a space robot”. In: *Acta Astronautica* 176 (2020), pp. 24–32.

[17] Y. She and S. Li. “An extra degree-of-freedom model for combined spacecraft attitude control with unilateral contact constraint”. In: *Acta Astronautica* 165 (2019), pp. 54–67.

[18] P. Huang, Y. Lu, M. Wang, Z. Meng, Y. Zhang, and F. Zhang. “Postcapture attitude takeover control of a partially failed spacecraft with parametric uncertainties”. In: *IEEE Transactions on Automation Science and Engineering* 16.2 (2018), pp. 919–930.

[19] P. Huang, M. Wang, Z. Meng, F. Zhang, Z. Liu, and H. Chang. “Reconfigurable spacecraft attitude takeover control in post-capture of target by space manipulators”. In: *Journal of the Franklin Institute* 353.9 (2016), pp. 1985–2008.

[20] S. Sastry and M. Bodson. *Adaptive control: stability, convergence and robustness*. Courier Corporation, 2011.

[21] J.-F. Shi, S. Ulrich, and A. Allen. “Spacecraft adaptive attitude control with application to space station free-flyer robotic capture”. In: *AIAA Guidance, Navigation, and Control Conference*. 2015, p. 1780.

[22] P. B. Koganti and F. E. Udwadia. “Dynamics and Precision Control of Uncertain Tumbling Multibody Systems”. In: *Journal of Guidance, Control, and Dynamics* 40.5 (2017), pp. 1176–1190. doi: 10.2514/1.G002212.

[23] J. Luo, C. Wei, H. Dai, Z. Yin, X. Wei, and J. Yuan. “Robust inertia-free attitude takeover control of postcapture combined spacecraft with guaranteed prescribed performance”. In: *ISA Transactions* 74 (2018), pp. 28–44. doi: 10.1016/j.isatra.2018.01.016.

[24] W. Cai, X. Liao, and D. Y. Song. “Indirect Robust Adaptive Fault-Tolerant Control for Attitude Tracking of Spacecraft”. In: *Journal of Guidance, Control, and Dynamics* 31.5 (2008), pp. 1456–1463. doi: 10.2514/1.31158.

[25] A. K. Sanyal, A. Fosbury, N. A. Chaturvedi, and D. S. Bernstein. “Inertia-Free Spacecraft Attitude Trajectory Tracking with Internal-Model-Based Disturbance Rejection and Almost Global Stabilization”. In: *Proceedings of the American Control Conference (ACC)*. St. Louis, MO, USA: IEEE, June 2009. doi: 10.1109/ACC.2009.5160039.

[26] A. K. Sanyal, A. Fosbury, N. A. Chaturvedi, and D. S. Bernstein. “Inertia-Free Spacecraft Attitude Tracking with Disturbance Rejection and Almost Global Stabilization”. In: *Journal of Guidance, Control, and Dynamics* 32.4 (2009), pp. 1167–1178. doi: 10.2514/1.41565.

[27] A. Weiss, I. Kolmanovsky, D. S. Bernstein, and A. K. Sanyal. “Inertia-Free Spacecraft Attitude Control Using Reaction Wheels”. In: *Journal of Guidance, Control, and Dynamics* 36.5 (2013), pp. 1425–1439. doi: 10.2514/1.58363.

[28] A. Weiss, X. Yang, I. Kolmanovsky, and D. S. Bernstein. “Inertia-Free Spacecraft Attitude Control with Reaction-Wheel Actuation”. In: *AIAA Guidance, Navigation, and Control Conference*. AIAA 2010-8297. Toronto, Ontario, Canada, Aug. 2010. doi: 10.2514/6.2010-8297.

[29] C. Liu, K. Shi, X. Yue, and Z. Sun. “Inertia-free saturated output feedback attitude stabilization for uncertain spacecraft”. In: *International Journal of Robust and Nonlinear Control* 30.13 (2020), pp. 5101–5121. doi: 10.1002/rnc.5044.

[30] J.-E. Slotine and M. Di Benedetto. “Hamiltonian adaptive control of spacecraft”. In: *IEEE Transactions on Automatic Control* 35.7 (1990), pp. 848–852.

[31] K. Lu and Y. Xia. “Finite-time attitude stabilization for rigid spacecraft”. In: *International journal of robust and nonlinear control* 25.1 (2015), pp. 32–51.

[32] I. Ali, G. Radice, and J. Kim. “Backstepping control design with actuator torque bound for spacecraft attitude maneuver”. In: *Journal of guidance, control, and dynamics* 33.1 (2010), pp. 254–259.

[33] K. H. Ang, G. C. Y. Chong, and Y. Li. “PID control system analysis, design, and technology”. In: *IEEE Transactions on Control Systems Technology* 13.4 (2005), pp. 559–576. doi: 10.1109/TCST.2005.847331.

[34] D. H. Kim and J. H. Cho. “A biologically inspired intelligent PID controller tuning for AVR systems”. In: *International Journal of Control, Automation, and Systems* 4.5 (2006). Retracted in 2011; see Retraction Notice DOI: 10.1007/s12555-011-0425-7, pp. 624–636.

[35] C. P. Bechlioulis and G. A. Rovithakis. “Robust adaptive control of feedback linearizable MIMO nonlinear systems with prescribed performance”. In: *IEEE Transactions on Automatic Control* 53.9 (2008), pp. 2090–2099. doi: 10.1109/TAC.2008.929402.

[36] Y. Hu, Y. Geng, B. Wu, and D. Wang. “Model-Free Prescribed Performance Control for Spacecraft Attitude Tracking”. In: *IEEE Transactions on Control Systems Technology* 29.1 (2021), pp. 165–179. doi: 10.1109/TCST.2020.2968868.

[37] A. K. Kostarigka and G. A. Rovithakis. “Prescribed performance output feedback/observer-free robust adaptive control of uncertain systems using neural networks”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 41.6 (2011), pp. 1483–1494. doi: 10.1109/TSMCB.2011.2154328.

[38] J. Na. “Adaptive prescribed performance control of nonlinear systems with unknown dead zone”. In: *International Journal of Adaptive Control and Signal Processing* 27.5 (2013), pp. 426–446. doi: 10.1002/acs.2322.

[39] Y. Li, S. Tong, L. Liu, and G. Feng. “Adaptive output-feedback control design with prescribed performance for switched nonlinear systems”. In: *Automatica* 80 (2017), pp. 225–231. doi: 10.1016/j.automatica.2017.02.005.

[40] J. Luo, C. Wei, H. Dai, and J. Yuan. “Robust LS-SVM-based adaptive constrained control for a class of uncertain nonlinear systems with time-varying predefined performance”. In: *Communications in Nonlinear Science and Numerical Simulation* 56 (2018), pp. 561–587. doi: 10.1016/j.cnsns.2017.09.004.

[41] Z.-S. Hou and Z. Wang. “From model-based control to data-driven control: Survey, classification and perspective”. In: *Information Sciences* 235 (2013), pp. 3–35. doi: 10.1016/j.ins.2012.07.014.

[42] Y. Jiang and Z.-P. Jiang. “Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics”. In: *Automatica* 48.10 (2012), pp. 2699–2704. doi: 10.1016/j.automatica.2012.06.096.

[43] H. Jiang, B. Zhou, D. Li, and G. Duan. “Data-driven-based attitude control of combined spacecraft with noncooperative target”. In: *International Journal of Robust and Nonlinear Control* 29.16 (2019), pp. 5801–5819. doi: 10.1002/rnc.4693.

[44] A. M. Zou, K. D. Kumar, Z. G. Hou, and X. Liu. “Finite-Time Attitude Tracking Control for Spacecraft Using Terminal Sliding Mode and Chebyshev Neural Network”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 41.4 (2011), pp. 950–963. doi: 10.1109/TSMCB.2010.2101592.

[45] C. Wei, J. Luo, H. Dai, Z. Bian, and J. Yuan. “Learning-based adaptive prescribed performance control of postcapture space robot-target combination without inertia identifications”. In: *Acta Astronautica* 146 (2018), pp. 228–242. doi: 10.1016/j.actaastro.2018.03.007.

[46] Y. She, J. Sun, S. Li, W. Li, and T. Song. “Quasi-model free control for the post-capture operation of a non-cooperative target”. In: *Acta Astronautica* 147 (2018), pp. 59–70. doi: 10.1016/j.actaastro.2018.03.041.

[47] F. L. Lewis and D. Vrabie. “Reinforcement Learning and Adaptive Dynamic Programming for Feedback Control”. In: *IEEE Circuits and Systems Magazine* 9.3 (2009), pp. 32–50. doi: 10.1109/MCAS.2009.933854.

[48] Q.-Y. Fan and G.-H. Yang. “Adaptive Actor–Critic Design-Based Integral Sliding-Mode Control for Partially Unknown Nonlinear Systems With Input Disturbances”. In: *IEEE Transactions on Neural Networks and Learning Systems* 27.1 (2016), pp. 165–177. doi: 10.1109/TNNLS.2015.2472974.

[49] R. S. Sutton and A. G. Barto. “Time-derivative models of Pavlovian reinforcement”. In: *Learning and computational neuroscience: Foundations of adaptive networks* (1988).

[50] Q. Wei and D. Liu. “Data-Driven Neuro-Optimal Temperature Control of Water–Gas Shift Reaction Using Stable Iterative Adaptive Dynamic Programming”. In: *IEEE Transactions on Industrial Electronics* 61.11 (2014), pp. 6399–6408. doi: 10.1109/TIE.2014.2301770.

[51] W. Guo, F. Liu, J. Si, D. He, R. Harley, and S. Mei. “Online Supplementary ADP Learning Controller Design and Application to Power System Frequency Control with Large-Scale Wind Energy Integration”. In: *IEEE Transactions on Neural Networks and Learning Systems* 27.8 (2016), pp. 1748–1761. doi: 10.1109/TNNLS.2015.2431734.

[52] A. Heydari. “Revisiting Approximate Dynamic Programming and Its Convergence”. In: *IEEE Transactions on Cybernetics* 44.12 (2014), pp. 2733–2743. doi: 10.1109/TCYB.2013.2291111.

[53] B. Luo, D. Liu, H. N. Wu, D. Wang, and F. L. Lewis. “Policy Gradient Adaptive Dynamic Programming for Data-Based Optimal Control”. In: *IEEE Transactions on Cybernetics* 47.10 (2017), pp. 3341–3354. doi: 10.1109/TCYB.2016.2623859.

[54] R. Kamalapurkar, P. Walters, J. Rosenfeld, and W. Dixon. “Approximate Dynamic Programming”. In: *Reinforcement Learning for Optimal Feedback Control: A Lyapunov-Based Approach*. Communications and Control Engineering. Cham: Springer, 2018, pp. 17–42. doi: 10.1007/978-3-319-78384-0_2.

[55] V. Konda and J. Tsitsiklis. “Actor-critic algorithms”. In: *Advances in neural information processing systems* 12 (1999).

[56] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. “Deterministic policy gradient algorithms”. In: *International conference on machine learning*. Pmlr. 2014, pp. 387–395.

[57] S. Fujimoto, H. Hoof, and D. Meger. “Addressing function approximation error in actor-critic methods”. In: *International conference on machine learning*. PMLR. 2018, pp. 1587–1596.

[58] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor”. In: *International conference on machine learning*. Pmlr. 2018, pp. 1861–1870.

[59] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. “Proximal policy optimization algorithms”. In: *arXiv preprint arXiv:1707.06347* (2017).

[60] M. Neunert, A. Abdolmaleki, M. Wulfmeier, T. Lampe, T. Springenberg, R. Hafner, F. Romano, J. Buchli, N. Heess, and M. Riedmiller. “Continuous-discrete reinforcement learning for hybrid control in robotics”. In: *Conference on Robot Learning*. PMLR. 2020, pp. 735–751.

[61] E. Puiutta and E. M. Veith. “Explainable reinforcement learning: A survey”. In: *International cross-domain conference for machine learning and knowledge extraction*. Springer. 2020, pp. 77–95.

[62] Y.-H. Cheng, B. Jiang, H. Li, and X.-D. Han. “On-Orbit Reconfiguration Using Adaptive Dynamic Programming for Multi-Mission-Constrained Spacecraft Attitude Control System”. In: *International Journal of Control, Automation and Systems* 17.4 (2019), pp. 822–835. doi: 10.1007/s12555-018-9308-5.

[63] L.-G. Gong, Q. Wang, and C.-Y. Dong. “Spacecraft Output Feedback Attitude Control Based on Extended State Observer and Adaptive Dynamic Programming”. In: *Journal of the Franklin Institute* 356.10 (2019), pp. 4971–5000. doi: 10.1016/j.jfranklin.2019.04.018.

[64] B. Pang, T. Bian, and Z.-P. Jiang. “Adaptive Dynamic Programming for Finite-Horizon Optimal Control of Linear Time-Varying Discrete-Time Systems”. In: *Control Theory and Technology* 17.1 (2019), pp. 73–84. doi: 10.1007/s11768-019-8168-8.

[65] A. Ghorbanpour. “Fixed Final Time Satellite Attitude Control with Thrusters Based on Dynamic Programming and Neural Networks”. In: *International Journal of Space Science and Engineering* 6.3 (2021), pp. 257–274. doi: 10.1504/IJSPACESE.2021.113885.

[66] J. G. Elkins, R. Sood, and C. Rumpf. “Adaptive Continuous Control of Spacecraft Attitude Using Deep Reinforcement Learning”. In: *Proceedings of the 2020 AAS/AIAA Astrodynamics Specialist Conference*. AAS 20–475. Lake Tahoe, USA (virtual), 2020.

[67] F. Vedant, J. T. Allison, M. West, and A. R. M. Ghosh. “Reinforcement Learning for Spacecraft Attitude Control”. In: *Proceedings of the International Astronautical Congress (IAC)*. IAC–19_C1_IP_4_x49857. Washington, DC, USA, 2019.

[68] Y. Zhou, E.-J. van Kampen, and Q. P. Chu. “Incremental Approximate Dynamic Programming for Nonlinear Adaptive Tracking Control with Partial Observability”. In: *Journal of Guidance, Control, and Dynamics* 41.12 (2018), pp. 2554–2567. doi: 10.2514/1.G003472.

[69] B. Gaudet, R. Linares, and R. Furfaro. “Deep Reinforcement Learning for Six Degree-of-Freedom Planetary Powered Descent and Landing”. In: *Advances in Space Research* 65.7 (2020), pp. 1723–1741. doi: 10.1016/j.asr.2019.12.030.

[70] K. Hovell and S. Ulrich. “Deep Reinforcement Learning for Spacecraft Proximity Operations Guidance”. In: *Journal of Spacecraft and Rockets* 58.2 (2021), pp. 254–264. doi: 10.2514/1.A34838.

[71] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. “Sim-to-real transfer of robotic control with dynamics randomization”. In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 3803–3810.

[72] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein. “Visualizing the loss landscape of neural nets”. In: *Advances in neural information processing systems* 31 (2018).

[73] R. Sullivan, J. K. Terry, B. Black, and J. P. Dickerson. “Cliff diving: Exploring reward surfaces in reinforcement learning environments”. In: *arXiv preprint arXiv:2205.07015* (2022).

[74] N. Rahn, P. D’Oro, H. Wiltzer, P.-L. Bacon, and M. Bellemare. “Policy optimization in a noisy neighborhood: On return landscapes in continuous control”. In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 30618–30640.

[75] J. Schneider, P. Schumacher, D. Häufle, B. Schölkopf, and D. Büchler. “Investigating the impact of action representations in policy gradient algorithms”. In: *arXiv preprint arXiv:2309.06921* (2023).

[76] O. Bastani, Y. Pu, and A. Solar-Lezama. “Verifiable reinforcement learning via policy extraction”. In: *Advances in neural information processing systems* 31 (2018).

[77] B. Hayes and J. A. Shah. “Improving robot controller transparency through autonomous policy explanation”. In: *Proceedings of the 2017 ACM/IEEE international conference on human-robot interaction*. 2017, pp. 303–312.

[78] S. Greydanus, A. Koul, J. Dodge, and A. Fern. “Visualizing and understanding atari agents”. In: *International conference on machine learning*. PMLR. 2018, pp. 1792–1801.

2

Visualizing Critic Match Loss Landscapes for Interpretation of Online Reinforcement Learning Control Algorithms

Reinforcement learning has proven its power on various occasions. However, its performance is not always guaranteed when system dynamics change and often relies on users' empirical experience. For reinforcement learning algorithms with an actor-critic structure, the critic neural network reflects the approximation and optimization process in the RL algorithm. Analyzing the critic network performance helps understand the algorithmic mechanism. To support systematic interpretation of such algorithms in dynamic control problems, this work proposes a critic match loss landscape visualization method for online reinforcement learning. The method constructs a loss landscape by projecting recorded critic parameter trajectories onto a low-dimensional linear subspace. The critic match loss is evaluated over the projected parameter grid using fixed reference state samples and temporal-difference targets, yielding a three-dimensional loss surface and a two-dimensional optimization path characterizing critic learning behavior. To extend analysis beyond visual inspection, quantitative landscape indices and a normalized system performance index are introduced for comparison across different training outcomes. The approach is demonstrated using the Action-Dependent Heuristic Dynamic Programming algorithm on cart-pole and spacecraft attitude control tasks. Comparative analyses across projection methods and training stages reveal landscape characteristics associated with stable convergence and unstable learning. The proposed framework enables qualitative and quantitative interpretation of critic optimization behavior in online reinforcement learning.

This chapter is based on the revised manuscript: J. Liu, J. Guo, and E. Gill, "Visualizing Critic Match Loss Landscapes for Interpretation of Online Reinforcement Learning Control Algorithms," submitted to *Acta Astronautica*, DOI: https://doi.org/10.48550/arXiv.2603.14535.

2.1. Introduction

Reinforcement learning algorithms have been a research hot spot in recent years. These algorithms have demonstrated impressive performance in areas such as robotics [1], game playing [2], navigation and control [3], and decision-making tasks [4].

For system control with uncertainties, reinforcement learning methods have also been widely applied [5]. For example, in Active Debris Removal (ADR) using robotic arms, the system becomes a new combined spacecraft system with uncertainties due to the uncooperative target capture. For combined spacecraft with system uncertainties, several works have been using reinforcement learning methods [6–8], which rely on the sampled data during the control process, rather than on model information.

In some control scenarios, the environment changes constantly. For example, in the above-mentioned ADR scenario, the combined spacecraft system can be an uncertain system with an unknown moving appendage, which causes the system dynamics to constantly change. As a result, a real-time RL agent that interacts with the environment is needed. This is where online reinforcement learning is applied. For example, the Adaptive Dynamic Programming (ADP) method is originally a method based on optimal control. With the actor-critic (AC) architecture, it can be regarded as one kind of online reinforcement learning method. Action-Dependent Heuristic Dynamic Programming (ADHDP), or Q learning, was introduced as a supplementary controller to guarantee a proper dynamic performance [9]. ADHDP is also used as the sole controller in the post-capture scenario with unknown system parameters [10].

Despite the successful applications of reinforcement learning control mentioned above, there are also possibilities that reinforcement learning algorithms do not perform as expected. Since RL algorithms are commonly trained on a specific fixed environment, it may work for one system, yet it fails to generalize to another [11]. In the extreme condition, when a single system parameter changes, a well-tuned RL algorithm may fail to work [12]. Therefore, the interpretation of the algorithm's performance is important for understanding the mechanism of the algorithm and for improving the performance of the algorithm.

The interpretation of the reinforcement learning algorithm largely relies on visualization techniques. Visualization of learning processes and control performance is one aspect for interpretation of RL algorithms. In many learning-based control studies, classical visualization techniques are adopted, such as learning curves, parameter evolution, and system performance trajectories, to illustrate convergence behavior and control effectiveness [13, 14]. To further support the interpretation of reinforcement learning behavior, visualization techniques have been developed to examine loss functions and optimization landscapes. Using visualization methods based on "filter normalization", the structure of neural loss functions is explored and the effect of loss landscapes on generalization [15]. The filter-normalized method is further used to visualize reward surfaces for popular reinforcement learning

environments [16]. By mapping between a policy and return, the return landscape is generated, which can navigate away from noisy neighborhoods and thus improve policy robustness [17]. The characteristics of the actor loss function are studied by visualizing the loss functions [18]. By visualizing the optimization surface of the Reacher and Walker-walk task implemented by Proximal Policy Optimization (PPO), the action representation's significant influence on the learning performance is demonstrated [19].

The above-mentioned works mainly use visualization techniques to explore reward landscapes and actor loss landscapes, and interpret algorithm performance from the perspectives of policy outcome or actor optimization. Reward surfaces describe how changes in policy parameters affect cumulative return, while actor loss landscapes reflect characteristics of policy update mechanisms. However, these visualizations do not directly reveal how the critic module in the actor–critic structure is optimized, nor do they visualize the geometry of the critic module. In the actor–critic structure, the critic module is applied to approximate the value function or cost function, and its approximation accuracy significantly influences or even governs the learning stability of the algorithm. Visualizing the critic behavior, therefore, helps to reveal the learning mechanism of the critic module and provides useful information for the interpretation of reinforcement learning algorithms.

In reinforcement learning with an actor–critic structure, the critic network is trained by minimizing the temporal-difference (TD) error step by step, and the approximation process is carried out through parameter updates. The approximation accuracy of the critic largely impacts the convergence behavior of the algorithm. The training of a critic network is a process to minimize the cost by updating the weights, i.e., the parameters used for approximation. During online training, both the TD target and the state distribution evolve as the policy changes, which makes the TD objective inherently moving and difficult to visualize as a single and well-defined surface. To enable interpretation of critic learning behavior under such conditions, we construct a critic match loss derived from the TD error by fixing the reference data and targets associated with a given policy. Since the TD error is the training signal that directly drives the critic parameter updates, the resulting critic match loss provides a meaningful representation of critic learning behavior that can be visualized as a well-defined loss landscape, which allows the critic optimization process to be interpreted from a geometric perspective.

In this work, we visualize the performance of the critic neural network for online reinforcement learning by constructing a critic match loss landscape. The critic weight parameters at the end of each training episode are recorded to keep track of the optimization path of critic weight during training. The corresponding state data and temporal-difference targets associated with selected policy references are used to reconstruct the critic match loss landscape over candidate weight parameters. These parameters are projected onto a principal component plane based on the optimization path of critic weight. Using the method above, the derived 3-D loss landscape and 2-D loss

path can qualitatively reveal the training evolution and expose local geometry under a given policy reference. It can also help to explain why a single RL algorithm converges in one system yet diverges in another.

The remainder of this paper is organized as follows. In Section 2, the critic loss landscape visualization method for online RL is introduced. The ADHDP algorithm is presented as a sample algorithm for the visualization technique. The quantitative indices for analyzing the loss landscape and system performance are given. In Section 3, using the cart-pole system and the spacecraft attitude system as the control object, the dynamic models of the two systems are introduced. The corresponding control results using ADHDP algorithms are shown. In Section 4, the performance of the ADHDP algorithm is interpreted using the critic match loss landscape visualization method. Comparisons are made across systems, across projection methods and selected policies. In Section 5, a conclusion is drawn.

2.2. Method

In this section, the visualization method for interpreting online RL algorithms is introduced. The ADHDP algorithm is also given, which is used as an object to be interpreted using the visualization method. Quantitative indices for quantitatively analyzing the loss landscapes with system performance are given.

2.2.1. Visualizing the loss function

Reinforcement learning uses neural networks as a tool to approximate functions. Normally, these neural networks approximate complex functions with many parameters. Visualizing the loss with respect to the network parameters can reveal the characteristics of the neural network. For example, flat regions indicate parameter settings where small perturbations have little effect on the loss, while sharp regions correspond to areas where small changes lead to large variations, highlighting local minima and saddle points. All of these interesting characteristics will be visible in the loss landscape. However, the large number of parameters makes the loss landscape visualization a multi-dimensional problem, which is difficult to visualize. To tackle this problem, the work [15] used a method called Contour Plots and Random Directions. The main idea of the method is that, since it's not possible to fully visualize how the loss changes with the full-dimensional parameters, two dimensions are selected to generate a 3-D loss landscape. The 3-D loss landscape is a projection slice of the full-dimensional loss landscape. To calculate the loss regarding the two selected dimensions, the following equation is used

$$f(\alpha, \beta) = \mathcal{L}(\boldsymbol{\zeta}^* + \alpha\boldsymbol{\delta} + \beta\boldsymbol{\eta}). \tag{2.1}$$

Here, $\boldsymbol{\zeta}^*$ is the chosen center point in the landscape graph while $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$ are the two directions chosen to visualize the landscape. The parameters α and β

are the distances from the chosen center point.

The equation could be understood in the following way. The multidimensional problem is reduced to a 3-D coordinate system. The loss is visualized on a 2-D subspace spanned by the chosen directions $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$. The 3-D plot shows the loss value f along the vertical axis, with α and β as coordinates on the horizontal plane.

Using the aforementioned Contour Plots and Random Direction method, the high-dimensional loss landscape can be visualized through a three-dimensional slice defined by two selected directions. Although this representation does not capture the complete geometry of the full parameter space, it provides an interpretable view of the local loss structure around the chosen point.

2.2.2. Action-dependent heuristic dynamic programming (ADHDP)

To illustrate how the critic loss landscape visualization method contributes to the interpretation of online RL methods, Action-dependent heuristic dynamic programming (ADHDP), a specific RL method, is used here as an example. ADHDP is a specific type of reinforcement learning method. In this method, the objectives of controlling and learning are separated and realized with two separate neural networks, which are the actor neural network and the critic neural network, as shown in Figure 2.1 [20]. The critic network is trained toward optimizing the total reward-to-go objective, which is the approximation of the cost function. The actor neural network is trained so that the critic network generates an ultimate objective of success, which is to minimize the cost.

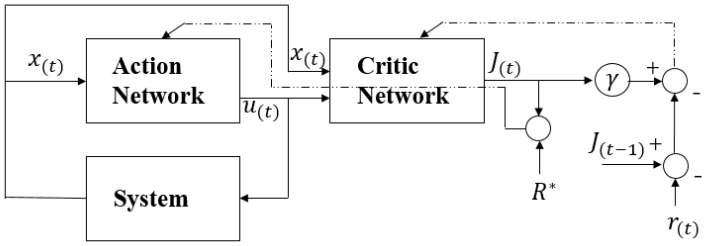


Figure 2.1: Structure of ADHDP

In ADHDP, the cost function $J(t)$ is expressed as

$$J_{(t)} = r(t+1) + \gamma r(t+2) + \cdots = \sum_{k=1}^{\infty} \gamma^{k-1} r(t+k). \tag{2.2}$$

Here, $r(t+1)$ denotes the reinforcement signal (reward) received by the system. k is the number of time steps from time instance t. The discount

factor γ determines the relative importance of future rewards in the cumulative return.

We define the prediction error e_c for the critic network as

$$e_c(t) = [r(t) + \gamma J(t)] - J(t-1). \tag{2.3}$$

The prediction error is also called TD error in reinforcement learning.

The set of weight parameters in the critic network is $\mathbf{w}_c$, as shown in Equation A.1. The weight update rule for the critic network is based on the gradient descent method.

in this chapter, the critic and actor networks are approximated using the Multilayer Perceptron (MLP) structure with one hidden layer. Detailed information about the ADHDP algorithm can be found in Appendix A.

2.2.3. Visualizing the critic match loss function for ADHDP

The ADHDP algorithm is trained based on Equation 2.3. The approximation precision of cost function J will influence the performance of the algorithm. The approximation error, TD error in Equation 2.3 is the loss for the critic network. Hence, visualizing the critic match loss landscape will reveal the approximation process of the cost function. From the landscape, we can observe the general trend of how the algorithm evolves during its update.

In this study, the following terminology is used to describe the training process. An *episode* denotes one complete simulation run of the controlled system starting from the initial state until termination. Within each episode, the system dynamics are propagated in discrete *simulation time steps* or *update steps*. After each simulation time step, the actor and critic neural networks are updated using the data collected at that step. The number of gradient update passes applied to the networks at each time step is referred to as the *epoch*. Therefore, one episode consists of multiple simulation time steps, and at each time step the networks are trained for a fixed number of epochs.

The critic network is implemented as a multilayer perceptron (MLP) with parameters denoted by $\mathbf{w}_c$. These parameters are updated at each time step, reflecting changes in the critic network during training. During online training the system is reset at the beginning of each episode while the network parameters continue from the previous episode. To track this evolution, the vector $\mathbf{w}_c$ is recorded at the end of each episode. This forms a sequence of critic weight snapshots $\{\mathbf{w}_c(k)\}_{k=1}^{N_{\text{snap}}}$, where N_{snap} denotes the total number of recorded episodes.

From Equation 2.1, two directions have to be selected to function as the two axes for landscape visualization. To select the two directions, the Principal Component Analysis (PCA) method is applied to generate the two orthogonal directions in the $\mathbf{w}_c$ vector group.

Let $\{\mathbf{w}_c(k)\}_{k=1}^{N_{\text{snap}}}$ denote the recorded critic weight snapshots at the end of each

episode, and define the centered data matrix

$$\mathbf{X} = \begin{bmatrix} (\mathbf{w}_c(1) - \bar{\mathbf{w}}_c)^\top \\ (\mathbf{w}_c(2) - \bar{\mathbf{w}}_c)^\top \\ \vdots \\ (\mathbf{w}_c(N_{\text{snap}}) - \bar{\mathbf{w}}_c)^\top \end{bmatrix}, \qquad \bar{\mathbf{w}}_c = \frac{1}{N_{\text{snap}}} \sum_{k=1}^{N_{\text{snap}}} \mathbf{w}_c(k). \tag{2.4}$$

The sample covariance matrix is computed as

$$\mathbf{C} = \frac{1}{N_{\text{snap}} - 1} \mathbf{X}^\top \mathbf{X}. \tag{2.5}$$

By eigen-decomposition,

$$\mathbf{C}\mathbf{v}_i = \lambda_i^{\text{PCA}} \mathbf{v}_i, \qquad \lambda_1^{\text{PCA}} \geq \lambda_2^{\text{PCA}} \geq \cdots, \tag{2.6}$$

and the first two eigenvectors are selected as the visualization directions,

$$\boldsymbol{\delta} = \mathbf{v}_1, \qquad \boldsymbol{\eta} = \mathbf{v}_2. \tag{2.7}$$

Here, N_{snap} denotes the number of recorded critic weight snapshots, $\mathbf{w}_c(k)$ represents the critic weight vector at episode k. $\bar{\mathbf{w}}_c$ is the mean of the recorded critic weights, which is introduced only for the PCA computation. It is not used as the reference point of the loss landscape, which is instead centered at the selected critic weight snapshot. $\mathbf{X}$ is the centered data matrix constructed from the weight trajectory, $\mathbf{C}$ is the sample covariance matrix, λ_i^{PCA} and $\mathbf{v}_i$ denote the eigenvalues and eigenvectors of $\mathbf{C}$, respectively. The first two eigenvectors $\mathbf{v}_1$ and $\mathbf{v}_2$ correspond to the two principal directions used to construct the visualization plane.

PCA is performed only on the recorded critic weight trajectory collected at the end of each episode. No additional candidate or grid-sampled weights are included in the PCA computation.

Now, we have two main orthogonal directions chosen to represent the $\mathbf{w}_c$ vector, which is $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$ in the equation. To generate the full loss landscape, rather than a one-dimensional loss curve, the parameters α and β are sampled over a grid. To calculate the loss corresponding to each combination of α and β in the coordinate system, training data should be used for the calculation. In a supervised machine learning scenario, the training data is a batch of static data. However, in online reinforcement learning, the training data are collected and updated over time, and thus differ across episodes. Therefore, to construct a well-defined loss landscape as a scalar field over the critic parameter space, the input data and corresponding targets must be fixed when scanning the parameter grid.

Here, we consider that the data collected at each episode of training is one batch of data. The batch data in one episode is generated by the corresponding policy of that episode. In this study, unless otherwise specified, the reference batch dataset is selected as the states collected at each simulation time step

of the final episode, and the corresponding targets are the temporal difference targets computed under the final policy. With the weight value of each weight grid, the input states data and the target value, the loss corresponding to each weight grid is calculated.

To construct the loss landscape, the reduced coordinates on the visualization plane must be mapped back to the original critic parameter space. For each selected grid point (α, β), the reduced coordinates are then mapped back to the original high-dimensional critic parameter space to reconstruct a full critic weight vector as

$$\tilde{\mathbf{w}}_c(\alpha, \beta) = \mathbf{w}_c^r + \alpha\boldsymbol{\delta} + \beta\boldsymbol{\eta}, \tag{2.8}$$

where $\mathbf{w}_c^r$ denotes the reference critic weight vector corresponding to the selected policy. For each reconstructed critic weight vector $\tilde{\mathbf{w}}_c(\alpha, \beta)$, the critic network is evaluated on a fixed reference dataset collected from a selected training episode. Let the reference dataset be denoted as $\{x_t^{\text{state}}, u_t^{\text{action}}, y_t^{\text{td}}\}_{t=1}^{N_{\text{ref}}}$, where x_t^{state} represents the system state at time step t, u_t^{action} denotes the corresponding control action, y_t^{td} is the temporal difference target computed under the reference policy, and N_{ref} is the number of samples in the reference dataset.

The critic match loss used for landscape construction is defined as

$$L_{\text{match}}(\alpha, \beta) = \frac{1}{N_{\text{ref}}} \sum_{t=1}^{N_{\text{ref}}} \left(J(x_t^{\text{state}}, u_t^{\text{action}}; \tilde{\mathbf{w}}_c(\alpha, \beta)) - y_t^{\text{td}} \right)^2, \tag{2.9}$$

where $J(\cdot; \tilde{\mathbf{w}}_c(\alpha, \beta))$ denotes the critic network output under the reconstructed critic weight parameters.

For each grid point (α, β) on the visualization plane, the reconstructed critic weight vector is used to evaluate the difference between the critic output and the corresponding TD target on the fixed reference dataset. The resulting critic match loss therefore characterizes the local geometry of the critic approximation error under the reference dataset. It provides insight into the optimization behavior of the critic during training. Evaluating this loss over the grid produces a well-defined scalar field over the two-dimensional parameter plane, which can be visualized as a three-dimensional loss landscape.

Hence, we derive a 3-D critic loss landscape which can depict the loss geometry around the final policy. By projecting the 3-D loss landscape to a 2-D contour with the same PCA plane, the recorded weight parameters formulate the optimization path of critic weights during training, which qualitatively shows the evolution of training relative to the landscape. From the description of the method, principally, data from any episode could be choosen to visualize the loss landscape generated by the policy generated from the corresponding episode.

This construction yields a well-defined scalar field over the critic parameters with the inputs and targets fixed when scanning the grid. It therefore provides an interpretable view of the local geometry under the final policy and a qualitative depiction of the training trend through the overlaid path. It

does not reproduce the exact per-episode temporal difference objective, which changes with the data and targets. But it serves as a performance index that reveals the geometry of the training landscape, which is relevant for explaining convergence or instability. Other reference batch datasets, such as those from intermediate training stages, can also be used to generate landscape snapshots, as illustrated in subsection 2.4.5.

2.2.4. Quantitative analysis of loss landscape

The critic match loss landscape is derived as in subsection 2.2.3 and gives a qualitative view of the optimization path. To demonstrate the method beyond visual inspection and enable objective comparison, we introduce three quantitative indices.

sharpness, *basin area*, and *local anisotropy* are introduced to depict the loss landscape quantitatively. They capture complementary aspects of the loss geometry around the final critic parameters.

Since loss scales differ across algorithms and systems, raw values are not directly comparable. Let $L(\alpha,\beta)$ be the loss on the PCA plane and (α^*,β^*) the final parameters with L^*. We define the relative surface

$$\Delta L(\alpha,\beta) = L(\alpha,\beta) - L^*, \tag{2.10}$$

and normalize it by the interquartile range (IQR) of ΔL over the grid to obtain a dimensionless surface $\tilde{L}$.

Sharpness Sharpness measures how fast the loss rises when moving away from the final point. For radius ϵ,

$$\mathrm{Sharp}_\epsilon = \max_{\vartheta\in[0,2\pi)} \tilde{L}(\alpha^* + \epsilon\cos\vartheta,\ \beta^* + \epsilon\sin\vartheta)\,. \tag{2.11}$$

A larger Sharp_ϵ value indicates higher local stiffness that descent can be fast with small steps, while the stable step-size margin narrows and sensitivity to noise increases. A smaller Sharp_ϵ value indicates a flatter, more tolerant neighborhood.

Basin area The basin area quantifies the extent of the low-loss set around the final point using

$$A_\rho = \mathrm{Area}\Big\{(\alpha,\beta) \;\Big|\; \tilde{L}(\alpha,\beta) \le \rho\Big\}. \tag{2.12}$$

A larger A_ρ implies wider robustness around the final point while a very small or non-closed set indicates a fragile or non-convergent situation.

Local anisotropy Local anisotropy captures directional imbalance near (α^*,β^*). A quadratic fit of $\tilde{L}$ in a small neighborhood yields a 2×2 Hessian H. We use the condition number

$$\log\kappa = \log\left(\frac{\lambda_{\max}(H)}{\lambda_{\min}(H)}\right), \tag{2.13}$$

where $\lambda_{\max}$ and $\lambda_{\min}$ are the largest and smallest eigenvalues of H. Large $\log \kappa$ indicates a narrow, ill-conditioned valley. It means that around the final point, one direction is steep while the other is flat, which makes progress sensitive to step size and update direction. Small $\log \kappa$ corresponds to more isotropic curvature and smoother updates.

In conclusion, sharpness shows local stiffness and basin area describes tolerance. These two indices address different questions and should be interpreted with caution. Sharp landscapes can offer strong local attraction yet require tight step-size control, while wide basins indicate robustness to parameter perturbations. Anisotropy complements both by indicating how "skewed" the valley is. Together, these indices provide a compact quantitative description of the landscape. Combined with overlaid 2D-optimization path of the weight, they support comparisons across runs and help explain training behavior.

2.2.5. System performance index

The critic match loss landscape will be analyzed quantitatively using the three indices in subsection 2.2.4. To relate the landscape geometry to the actual control performance of a dynamic system, a scalar system performance index is introduced.

To enable consistent comparison across different systems, each policy is evaluated on a fixed horizon H_{eval} using the same stage cost $c(x_t, u_t)$ as in training, which corresponds to the reinforcement signal $r(t)$ as in Equation 2.2. For performance evaluation, this instantaneous cost is normalized by a system-dependent upper bound, such that $c(x_t, u_t) \in [0, 1]$. Then we have

$$J_{H_{\text{eval}}} = \sum_{t=0}^{H_{\text{eval}}-1} \gamma^t \, c(x_t, u_t). \tag{2.14}$$

If a failure occurs, such as states exceeding predefined constraints at time t_{fail}, the rollout is continued to the horizon H_{eval} with a fixed worst-case penalty

$$c(x_t, u_t) = c_{\max}, \qquad t \geq t_{\text{fail}}, \tag{2.15}$$

where $c_{\max} = 1$ corresponds to the maximum normalized stage cost.

Since $c(x_t, u_t) \in [0, 1]$ is by construction, the discounted cost admits a natural upper bound. The performance index is therefore normalized as

$$\tilde{J}_{H_{\text{eval}}} = \frac{J_{H_{\text{eval}}}}{\sum_{t=0}^{H_{\text{eval}}-1} \gamma^t} \in [0, 1]. \tag{2.16}$$

The variables appearing in the above formulation are defined as follows. The state vector x_t represents the system state at time step t, and u_t denotes the control input applied at the same time step. The function $c(x_t, u_t)$ is the instantaneous stage cost used for performance evaluation, which is normalized such that $c(x_t, u_t) \in [0, 1]$. The parameter $\gamma \in (0, 1]$ denotes the discount factor

that weights future costs. The evaluation is performed over a fixed horizon H_{eval}, and J_H represents the cumulative discounted cost over this horizon. The normalized performance index $\tilde{J}_H$ rescales the cumulative cost by the maximum possible discounted sum to ensure $\tilde{J}_H \in [0, 1]$. If a failure occurs at time t_{fail}, the stage cost is set to the worst-case value $c_{\max}$ for all subsequent time steps until the end of the horizon.

From this definition, a well-controlled behavior yields a small $\tilde{J}_H$, whereas early failure results in a large $\tilde{J}_H$ due to the accumulated penalty. This single index enables direct comparison of system performance across tasks and remains meaningful even when training diverges, providing a unified basis for relating control performance to the quantitative properties of the loss landscape.

2.3. ADHDP control result for system with uncertainties

In this section, the cart-pole system and the spacecraft attitude system are introduced. The control results of using the ADHDP algorithm are shown. The results will be used as examples for using the critic match loss landscape to interpret the online RL algorithm. Multiple independent simulation runs were conducted for each system. For visualization and analysis, representative runs were selected to illustrate typical convergent and divergent learning behaviors.

2.3.1. Dynamics and control for cart-pole system

Consider the physical model shown in Figure 2.2. A cart is positioned on a track running horizontally and an inverted pendulum is attached to the cart with a hinge that allows rotation around pivot point P in the $h - y$ plane only, i.e. the pendulum is free to move horizontally. The pendulum's initial position is vertical, and an external variable horizontal force is applied to the cart to maintain the pendulum in a balanced and upright position.

The model is considered to represent an unstable system. An external control force is required to keep the pendulum balanced, as opposed to a downward-hanging pendulum in which the force of gravity results in a stable oscillation about the downward vertical. An ideal pendulum is assumed in which all of the pendulum's mass is located at a single point (Q) at the end of a massless rigid rod.

The control goal of the cart-pole system is that by "pushing the cart to the right" or "pushing the cart to the left," the pendulum can stay in its upright position. Since the force is applied by pushing the cart either to the left or to the right, the control input to the cart-pole system is discrete with a constant magnitude. The actor network in ADHDP produces a continuous control signal, and the sign of this output is used to determine the direction of the applied force.

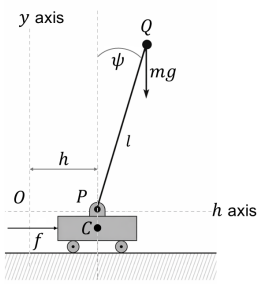


Figure 2.2: Cart-pole system as a benchmark problem for ADHDP

The dynamics and kinematics of the system are described with the following equations [21].

$$\ddot{\vartheta} = \frac{g \sin\psi + \cos\psi \left[\frac{-f - ml\dot{\psi}^2 \sin\psi + \mu_c \operatorname{sgn}(\dot{h})}{m_c + m} \right] - \frac{\mu_p \dot{\psi}}{ml}}{l \left[\frac{4}{3} - \frac{m\cos^2\psi}{m_c + m} \right]}, \tag{2.17}$$

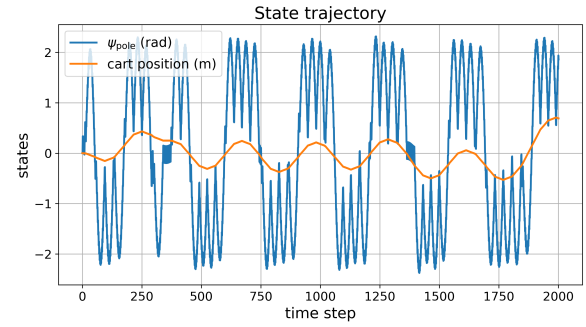
$$\begin{aligned} \ddot{h} &= \frac{f + ml\left[\dot{\psi}^2 \sin\psi - \bar{\psi}\cos\psi\right] - \mu_c \operatorname{sgn}(\dot{h})}{m_c + m} \\ g &= -9.8\,\mathrm{m/s^2}, \end{aligned} \tag{2.18}$$

where ψ is the angular placement of the pendulum, and h is the horizontal placement of the cart, as indicated in Figure 2.2. f is the force that is applied to the cart, m is the mass of the pendulum, m_c is the mass of the cart, μ_p and μ_c are the friction coefficients for the pendulum and the cart, and l is the length of the pendulum.

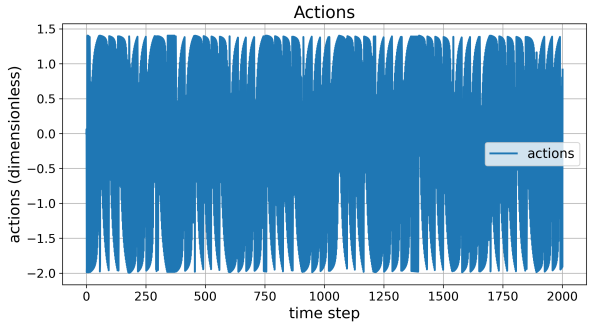
The ADHDP algorithm is first applied to the cart-pole dynamic system. Then, the critic loss landscape visualization method is applied to reveal the training process of the algorithm.

For the ADHDP control of the cart-pole system, the state vector is $[\psi, \dot{\psi}, h, \dot{h}]$, and the control input is $[f]$. The input to the critic neural network is the combination of the state vector and the control input. The input to the actor neural network is the state vector. The MLP structure with one hidden layer is chosen to approximate the critic and actor neural networks. The number of neurons of the hidden net is 6 and 4, respectively. The actor and critic learning rates are set to 1×10^{-3} and 1×10^{-2} respectively. Training is conducted over 100 episodes. The ADHDP algorithm is trained in an online manner, where at each dynamic simulation time step the current state is sampled and used immediately to compute the temporal-difference error and update the actor and critic networks. No minibatch training is employed, which is equivalent to a batch size of one. Network weights are initialized using uniformly distributed random values. No explicit exploration noise is added to the actor output in

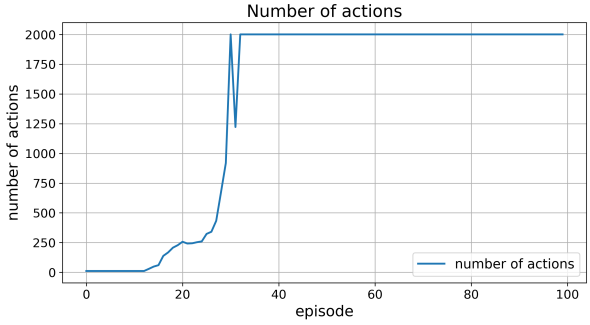
the baseline ADHDP implementation as shown in Appendix A, in order to avoid introducing additional stochasticity that could confound the interpretation of critic optimization behavior.



(a) State trajectory



(b) Actions

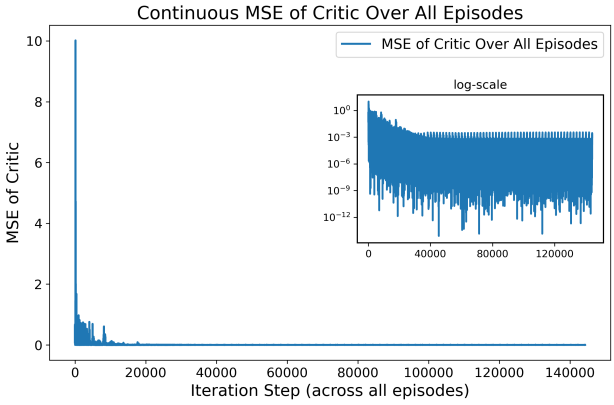


(c) Number of actions

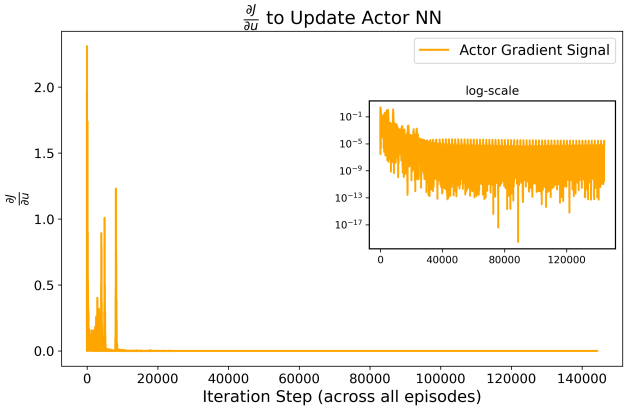
Figure 2.3: Cart-pole control results

Figure 2.3 shows a successful simulation result of using ADHDP to control the cart-pole system. Figure 2.3c shows how many actions the system can take during each episode before it diverges in 100 episodes of running

the simulation. Figure 2.3a shows the evolution of the states in the 100^{th} episode, which can be maintained at the desired range. Figure 2.3b shows the corresponding actions generated by the actor neural network. The system takes the sign of these actions as the direction of the applied force.



(a) Critic loss



(b) Actor gradient w.r.t. J

Figure 2.4: Critic and actor NN evolution process with time, cart-pole system

Figure 2.4a shows the critic match loss during training. It can be seen that, as the training continues, the critic match loss decreases to around zero. It means that the critic network is converging. Figure 2.4b shows the evolution of $\partial J(t)/\partial u(t)$.This item is used to update the actor neural net, as shown in Equation A.15. As the training continues, the gradient decreases to around zero. This indicates that the policy update becomes very small and the actor network reaches a stationary point. Since the achieved control performance is not globally optimal, the solution corresponds to a sub-optimal equilibrium.

2.3.2. Attitude dynamics and control for spacecraft with unknown inertia

In subsection 2.3.1, the ADHDP method has been applied to the benchmark cart-pole system. In this part, the results of applying the algorithm to the spacecraft attitude control system, a more complex problem with unknown inertia, will be studied.

Following the benchmark cart-pole system, the combined spacecraft dynamic system will be introduced in this part. From the literature review in section 2.1, the combined spacecraft in the ADR scenario is a system with uncertainties. In order to progressively test the performance of the algorithm on the control of the post-capture scenario, the combined spacecraft model with the lowest level of uncertainty is considered in the work, which is the uncertainty of the unknown inertial parameter. Based on this, the following assumptions are made. The target is captured tightly by rigid robotic manipulators on the rigid service spacecraft. The joints of spacecraft manipulators will be locked once the target spacecraft is captured [22]. The target is rigid and uncooperative without control capability. Under the given assumptions, the combined spacecraft is taken as a single rigid spacecraft with unknown inertia parameters.

In the present study, the reinforcement learning algorithm is trained in simulation rather than directly on the physical spacecraft. The training process provides an initial stabilizing control policy represented by the actor network. During operation, the learned policy is deployed as a feedback controller that generates control commands based on the observed spacecraft states. Pre-training the controller in simulation improves safety and stability, since learning does not need to start from an untrained policy when applied to the real system. During this simulation-based training stage, the critic match loss landscape can be used to analyze the evolution of the critic optimization and the corresponding control behaviour. This provides an interpretable view of the learning process before the controller is applied to the physical system.

The kinematic and dynamic equations of combined spacecraft in the inertia principal axis frame of combination use Euler angles. In this chapter, a body-fixed reference frame $\mathcal{B}$ is defined, located at the center-of-mass of the combined spacecraft. Its coordinate axes are aligned along the principal directions of the spacecraft. In addition, an Earth Centered Inertial (ECI) coordinate frame $\mathcal{N}$ is defined, with unit vectors $\{\overline{\boldsymbol{n_1}}, \overline{\boldsymbol{n_2}}, \overline{\boldsymbol{n_3}}\}$.The $\overline{\boldsymbol{n_1}}$ axis is aligned with the mean equinox. The $\overline{\boldsymbol{n_3}}$ axis is aligned with the Earth's rotation axis or the celestial North Pole, and the $\overline{\boldsymbol{n_2}}$ complies with the right-handed system.

For the rotation from frame $\mathcal{N}$ to frame $\mathcal{B}$ with the sequence of 3-2-1, the kinematics of the combined spacecraft are given below. In the scenario for the combined spacecraft stabilization, the assumption is that the initial attitude states of the combined spacecraft are not expected to deviate significantly from zero degrees, with the final control goal as zero degrees. Hence, the combined spacecraft here is represented with attitude angles, which will not

suffer from the singularity problem under the given scenario.

$$\begin{bmatrix} \dot{\partial}_1 \\ \dot{\partial}_2 \\ \dot{\partial}_3 \end{bmatrix} = \frac{1}{\cos \partial_2} \begin{bmatrix} \cos \partial_2 & \sin \partial_1 \sin \partial_2 & \cos \partial_1 \sin \partial_2 \\ 0 & \cos \partial_1 \cos \partial_2 & -\sin \partial_1 \cos \partial_2 \\ 0 & \sin \partial_1 & \cos \partial_1 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}. \tag{2.19}$$

The attitude dynamics is given as

$$\bar{M} = \hat{J}_{sc} \cdot \dot{\bar{\omega}} + \bar{\omega} \times \hat{J}_{sc} \cdot \bar{\omega}. \tag{2.20}$$

Here, ∂_1, ∂_2, and ∂_3 represent the three attitude angles of the spacecraft, and ω_1, ω_2, and ω_3 are the corresponding angular velocities. $\bar{M}$ denotes the control torque applied to the attitude system. $\hat{J}_{\mathrm{sc}}$ is the inertia matrix of the spacecraft, which is assumed to be unknown.

The reward function is given in the form of a quadratic function of state errors and control input, as

$$r(x, u) = x^T P x + u^T Q u, \tag{2.21}$$

where $x = [e_{\partial_1}, e_{\partial_2}, e_{\partial_3}, e_{\omega_1}, e_{\omega_2}, e_{\omega_3}]$ represents the error between the observed and desired states. For the attitude stabilization task, the desired states are all equal to zero. During this simulation, value of P is 0.01, value of Q is set to zero to reduce the complexity of the learning problem.

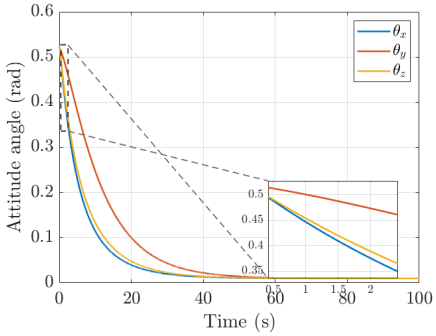
Regarding the inertia parameters, in practical ADR missions the inertia of the combined spacecraft is generally uncertain because the mass distribution of the captured target is unknown. In this work, the control problem is therefore formulated as stabilizing a rigid spacecraft with unknown inertia. The reinforcement learning algorithms used are model-free and do not require knowledge of the inertia matrix or other system parameters. For simulation, however, the dynamics must be generated with a specific inertia matrix, so a fixed inertia value is used to produce state transitions. This value is only used within the simulation and is not provided to the learning algorithm. The inertial parameter of the combined spacecraft is described with the matrix $\hat{J}_{sc}$ with value $[1\ 0.1\ 0.1;\ 0.1\ 0.1\ 0.1;\ 0.1\ 0.1\ 0.9]\ \mathrm{kg\,m^2}$.

As a reference for the performance of the ADHDP, we start by applying standard PD control to the problem. A disturbance torque of 0.01 N is added to the system. The control gains K_p and K_d are selected as $\mathrm{diag}(1, 1, 1)$ and $\mathrm{diag}(10, 10, 10)$, respectively. The gains are selected through manual tuning to provide a stable baseline response for comparison with the ADHDP controller. The results are shown in Figure 2.5.

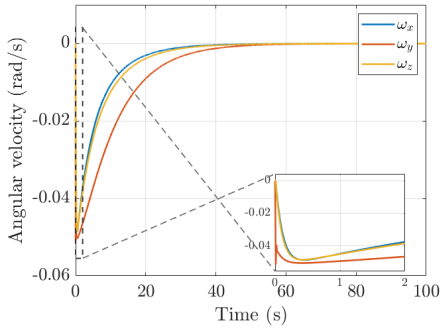
As can be seen from Figure 2.5, the system can be stabilized in the given 100 seconds. And the control torque does not exceed 1 Nm.

The ADHDP algorithm is applied to the above-defined spacecraft attitude dynamic system. Then the critic match loss landscape visualization method is applied to reveal the training process of the algorithm.

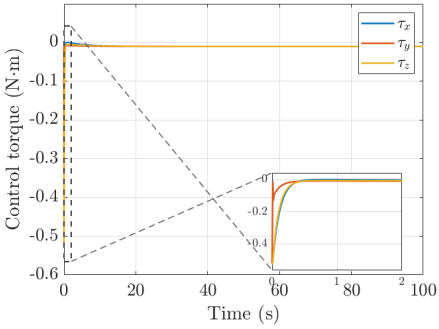
To simplify the ADHDP control problem, no external disturbances are considered, and the system is initialized at the desired equilibrium state, which



(a) Attitude angle



(b) Angular rate



(c) Control torque

Figure 2.5: Results with PD control

is zero. For the ADHDP control of the spacecraft attitude dynamic system, the state vector is $[\vartheta_1, \vartheta_2, \vartheta_3, \omega_1, \omega_2, \omega_3]$, and the control input is u. The input to the critic neural network is the combination of the state vector and the control input. The input to the actor neural network is the state vector. The MLP structure with one hidden layer is chosen to approximate the critic and actor neural networks. The number of neurons of the hidden net is 10 for both neural networks.

The hyperbolic tangent function is used as the activation function for the hidden layer in both actor and critic neural networks to add nonlinearity to the approximation process. For the selection of the activation function for the output layers of the neural network, the linear function is selected as the output function of the critic network, considering the range of the linear function and the cost function. The hyperbolic tangent function is used as the output activation of the actor network, since its range sufficiently accommodates the required control torque, which is less than 1 Nm as shown in Figure 2.5c. The actor and critic neural networks are initialized using the default initialization of the linear layers.

The actor and critic learning rates are set to 1×10^{-3} and 1×10^{-2} respectively. The critic learning weight is adaptively adjusted during training using a multiplicative scaling ratio of 1.01, with an upper bound of 1.2. The algorithm is trained in an online manner following the same update scheme as the cart-pole case in subsection 2.3.1. Parameter updates are performed at each simulation time step, corresponding to a batch size of one, and no explicit exploration noise is added.

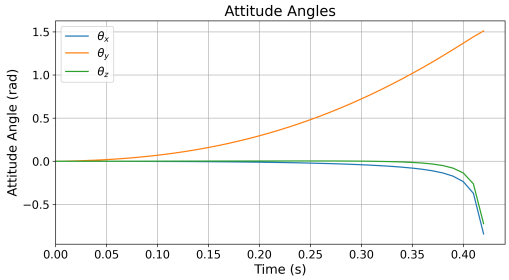
Training is conducted over 300 episodes. For each episode, using the PD control results in Figure 2.5 as a reference, the boundary for simulation time step is set for 100 seconds. For each time step in the episode, the actor and critic networks are trained in 30 epochs. To efficiently train the algorithm, a limit is set to the states of the system, which means that once the system starts to diverge and the states exceed the limit, the current episode will terminate. The limit can be set according to the simulation scenario. For example, a certain value can be given to the angular rate as the limit. For diverging runs, the final episode used for loss landscape construction corresponds to the last executed episode before training termination, which may be an early-terminated episode due to state divergence.

Figure 2.6 is the control result of applying the ADHDP algorithm to the spacecraft attitude system. Instead of stabilizing the attitude angles and angular rates to zero, the attitude angles, angular rates, and control torques all display divergent behavior. The algorithm doesn't manage to successfully control the system.

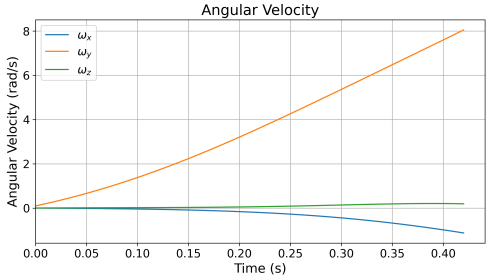
Figure 2.7 shows the critic match loss and the evolution of $\partial \mathrm{J}(t)/\partial \mathrm{u}(t)$ during the training process of the ADHDP algorithm in the spacecraft attitude control scenario. The critic loss remains low during the early phase of training, followed by a series of sharp spikes in the middle phase. In the final phase, the loss stabilizes again at a relatively low level, significantly below the peak values observed earlier. The actor gradient $\partial \mathrm{J(t)}/\partial \mathrm{u(t)}$ uses critic loss for calculation. So a similar trend is seen in Figure 2.7b: a gradual increase leading to several pronounced peaks during the middle phase, followed by smaller and more stable gradient values toward the end. However, when the training reaches the stop limit, the actor loss is still high, with an increasing trend, which is definitely indicating divergence.

2.4. Critic match loss landscape visualization for ADHDP algorithm

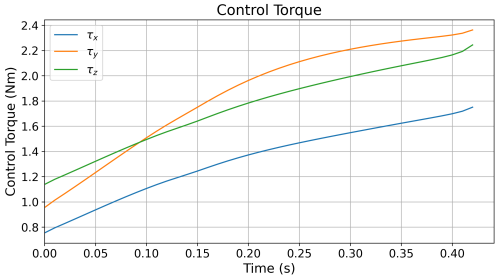
In this section, the results of using the visualization method on the ADHDP algorithms are shown. The performance of the algorithms will be analyzed using the critic match loss landscape. In the experiment, the loss landscape is visualized using the raw critic loss to preserve its original geometry, while a linear scaling is applied when computing the landscape quantitative indices to ensure numerical comparability across different control systems.



(a) Attitude angle



(b) Angular rate

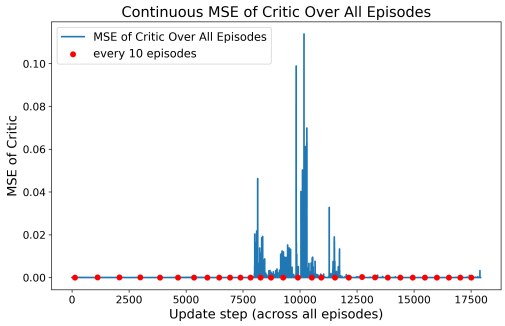


(c) Control torque

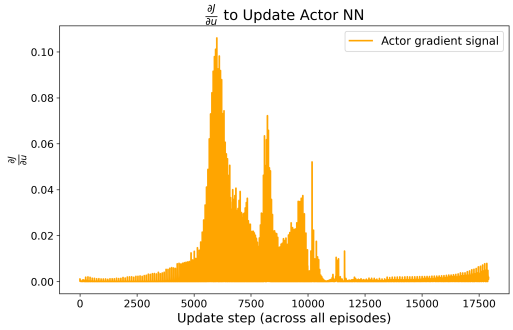
Figure 2.6: Spacecraft control results with ADHDP

2.4.1. Critic match loss landscape visualization for cart-pole ADHDP control under final policy

Figure 2.8a shows how the critic loss landscape changes with weight of the critic NN. According to section 3.1, the critic network is a MLP with one hidden layer. The combined weight matrix is a vector with multi-dimensions. After PCA is applied, the dimension of the matrix is recuded to two dimensions, which are illustrated by the x and y axes. The z axis indicates the loss across the weight grid on the x and y axes. The figure then shows the loss trend under different value combinations of the weight vector. Since the usage of the weight is to approximate the loss function, the 3-D loss landscape can be used to analyze

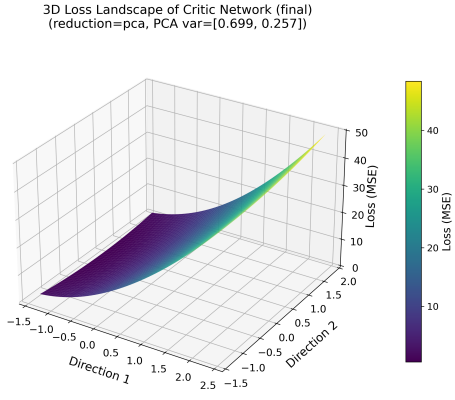


(a) Critic loss

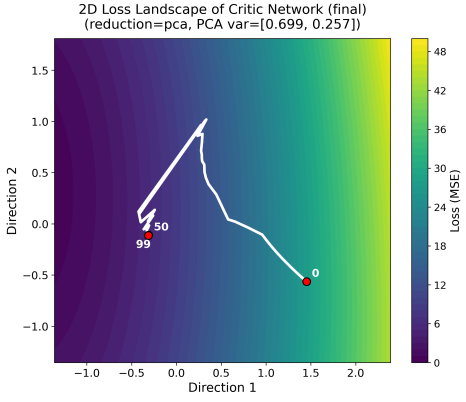


(b) Actor gradient w.r.t. J

Figure 2.7: Critic and actor NN evolution process with time, spacecraft system



(a) 3-D loss of cart-pole ADHDP control under final policy



(b) 2-D loss path of cart-pole ADHDP control under final policy

Figure 2.8: 3-D and 2-D loss landscape of cart-pole ADHDP control under final policy

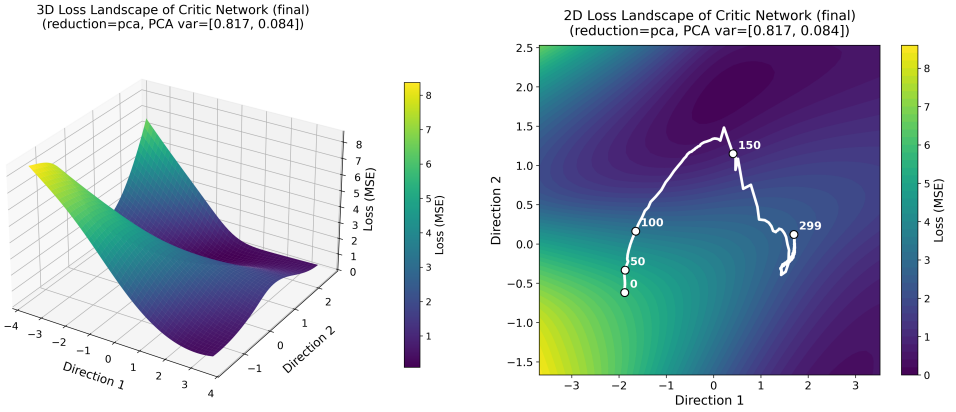
the characteristics of the algorithm revolution. From the example of using ADHDP to control the cart-pole system, it can be seen that the training brings a smooth loss landscape. Under this smooth loss landscape, the training process produces an accurate approximation of the loss function. This approximation of the cost function supports successive updates of the weight parameters. For the convergent cart-pole case shown in Figure 2.8a, the first two principal components explain 69.9% and 25.7% of the variance of the critic weight trajectory, respectively. Together, the first two principal components capture 95.6% of the total variance, indicating that the dominant directions of critic parameter evolution are well represented in the two-dimensional visualization plane.

Figure 2.8b is a 2-D projection of the 3-D loss landscape. The line in the projection indicates the update path of the weight during training. The beginning of the weight updates from a value that results in a relatively large critic match loss. As the training continues, the loss is optimized following the gradient descent of the weights. Finally, the weight update converges to a value that results in a low critic match loss. To illustrate the update trajectory, three representative points are marked along the 2-D path, corresponding to the beginning, a middle stage, and the final stage of training. It can be observed that the middle point is much closer to the final point than to the initial point. This indicates that a large portion of the loss reduction occurs in the early stage of training, while the updates become smaller as the training progresses. This pattern is consistent with gradient-based optimization, where large initial gradients drive fast loss reduction, followed by smaller updates near convergence. When the update stops, the process doesn't necessarily reach the global optimum, but it settles at a local minimum where the critic match loss becomes small.

In this convergent cart-pole case, the small critic loss is consistent with the stable control performance observed in the simulation. This point is referred to as a sub-optimal solution, which is common in online learning where exact optimization is difficult to achieve. This pattern corresponds with Figure 2.4a, which depicts the critic loss changes with update steps. The critic loss changes steeply in the early steps. After the sudden decrease, the critic loss stays around the convergent value, which indicates the weight behaviour around the sub-optimal point.

2.4.2. Critic match loss landscape visualization for spacecraft attitude control under final policy

Figure 2.9 illustrates the 3-D critic match loss landscape and the 2-D loss path resulting from applying the ADHDP algorithm to the spacecraft attitude system. Note that for this diverging case, the loss landscape is constructed using the final executed episode, which terminates early due to state divergence. From Figure 2.9a, it can be seen that the loss landscape based on the final policy yields a terrain which has two peak-like structure and two bowl-shaped regions.



(a) 3-D loss landscape

(b) 2-D loss landscape

Figure 2.9: 3-D and 2-D loss landscape of spacecraft attitude ADHDP control under final policy

The optimization path of the critic weight travels from peak to one of the bowls and ends in the other bowl, which corresponds to the fluctuations in loss values in Figure 2.7a. For the divergent spacecraft attitude control case shown in Figure 2.9a, the first two principal components explain 81.7% and 8.4% of the variance of the critic weight trajectory, capturing 90.1% of the total variance. The PCA results further indicate that the critic parameter updates exhibit strong anisotropy, with the first principal component accounting for 81.7% of the total variance. This implies that the evolution of the critic parameters during training is dominated by updates along a single principal direction in the parameter space. If this dominant update direction does not correspond to a descent direction of the true cost function, which may arise from unstable or rapidly varying TD target, the accumulated updates may drive the learning process toward divergence rather than convergence. This behavior is consistent with the highly anisotropic and non-convex structures observed in the final critic match loss landscape.

2.4.3. Cross-system comparison

In this subsection, the quantitative analysis of the critical match loss landscape will be shown by comparing the landscapes' quantitative indexes of the two systems. The quantitative analysis of the loss landscape will also link with the system performance using the performance index in subsection 2.2.5.

As shown in Figure 2.9a, the critic match loss landscape of the ADHDP algorithm for the spacecraft system is substantially more complex than that of the cart-pole system in Figure 2.8a. While the cart-pole case shows a smooth, single-slope surface, the spacecraft landscape contains multiple peaks and valleys, indicating a more challenging optimization process. Table 2.1 shows

the quantitative description of the critic match landscapes using sharpness, basin area and local anisotropy. The normalized cost of the two system is also shown in the system, which enables the link between the characterisitcs of the loss landscapes and system performances.

Table 2.1: Comparison of ADHDP performance on different environments

	Cart-pole system	**S/C attitude system**
Control results	stable	unstable
Landscape shape	Sloped	non-convex
Sharpness Sharp_ϵ	0.341747	0.267753
Basin area A_ρ	4.196473	20.596623
Local anisotropy $\log \kappa$	1.445154	2.651535
Normalized cost $\tilde{J}_H$	0.001029	0.148814

For cart-pole, the controller converges. The landscape is essentially a single slope with a clear descent direction. Table 2.1 shows the quantitative indices of the loss landscape of cart-pole ADHDP control. Sharpness is relatively high, the basin area is small, and local anisotropy is low. This pattern is consistent with a monotone surface. The large sharpness indicates a stiff neighborhood along the slope around the final point. The small basin area A_ρ shows a single tilted plane crossing the threshold only in a small region with low loss. The curvature around the final point is nearly isotropic with small $\log \kappa$. Under the same evaluation process, the system performance index is near zero, indicating successful control.

For the spacecraft (S/C) attitude case, the controller is unstable. The landscape presents multiple peaks and valleys and the path oscillates. Quantitatively, sharpness is small, the reported basin area is large, and local anisotropy is higher. The small sharpness reflects the lack of a single steep descent direction near the final point, while the large A_ρ is caused by several shallow low-loss patches connected under the threshold in a non-convex surface, which does not imply robustness of the solution. The larger anisotropy indicates skewed curvature and narrow passages between valleys. If we combine the three indices and the observation of the loss landscape, it can be seen that, the algorithm terminates in a final point with a large and flat area yet with high local anisotropy, while the loss value is still high. Such a region makes it difficult for the optimization process to escape toward more favorable directions. The corresponding finite-horizon performance index is also larger, matching the observed instability.

From the analysis of the landscapes of the cart-pole and spacecraft attitude control system, it can be seen that the indices should therefore be interpreted in combination. Sharpness captures local stiffness. Basin area reflects the extent of low-loss regions but can inflate on fragmented landscapes. And anisotropy shows directional difficulty. Their joint pattern aligns with the system performance index.

The differences in the observed loss landscapes are related to the inherent

dynamics and dimensionality of the control systems. The number of states and the dimensions of the control input are different for the cart-pole system and the spacecraft attitude system. For the cart-pole system, the number of states is 4, which are the horizontal placement and angle placement, and their derivatives. The dimension of the control input is 1. For the combined spacecraft system, the number of states is 6, which are the three attitude angles and their derivatives. The dimension of the control input is 3, which is the control torques in three directions. In the present implementation, the number of states and the dimensions of the control input determine the size of the input layer for the critic and actor networks. A higher-dimensional input leads to a larger number of network parameters that need to be trained and updated, which means that more parameters need to be trained and updated. As can be seen, with more states and dimensions of control, the structure of the neural network of the combined spacecraft system is more complicated than that of the cart-pole system.The differences in the loss landscape reflect the relative difficulty of optimizing the J cost function and training the algorithm for the two systems.

Figure 2.9b shows the 2-D projection of the 3-D critic match loss landscape. Different from Figure 2.8b, of which the weight directly brings the loss from a larger value to a smaller value, the critic weight updates in the spacecraft attitude system exhibit a less stable pattern. The trajectory begins at a point associated with a relatively high critic loss and proceeds through a series of local minima, indicating frequent shifts in the optimization path rather than smooth convergence. During the journey of traveling between different local minima points, the critic loss value experiences spiky changes, which correspond with Figure 2.7a.

However, it should be noted that the 2-D loss path does not strictly represent the critic loss at every update. The plot in Figure 2.7a provides the actual critic loss values recorded at each training step. In contrast, the critic match loss landscape is constructed by fixing a reference batch, namely the states from the final training episode and the corresponding temporal difference targets computed at the final episode, and then evaluating the critic match loss over candidate weights. Because online reinforcement learning evolves both data and targets over time, the landscape cannot reproduce every transient change.

This also raises an important question: if learning becomes unstable or fails to converge, does the loss landscape still provide meaningful information? In this work, the critic match loss landscape is not intended to reproduce the instantaneous critic loss during training. Instead, it evaluates candidate critic parameters on a fixed reference batch, which allows the geometry of the parameter space to be examined under a consistent objective. Even when learning is unstable, the resulting landscape still reflects how different critic parameter configurations approximate the value function for the same reference data. Therefore, the landscape does not assess the accuracy of the critic itself, but rather reveals the optimization geometry encountered by the algorithm. In unstable cases, this geometry often exposes fragmented low-loss

regions, skewed curvature, or weak descent directions, which helps explain the observed oscillations and sensitivity during training.

Considering the misalignment between the critic match loss landscape and critic loss curve with evolution, one naturally arrives at the question: why does the loss landscape still offer meaningful insight?

Taking the cart-pole system as an example, the ADHDP control yields a convergent and successful result. The cost function is well approximated by the final critic, and the stored sequence of weights traces the optimization path toward a broad low region of the landscape. When the grid is evaluated on the final episode data and final episode targets, weights corresponding to earlier stages produce higher loss, while weights near the final solution produce lower loss, which appears as a darker region in the contour.

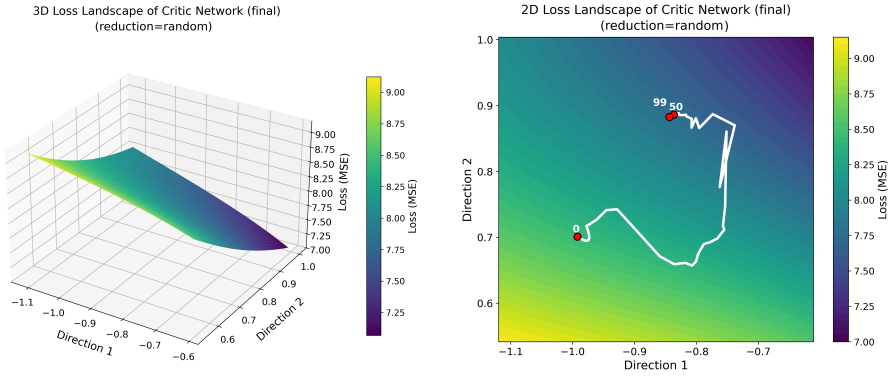
This behaviour is consistent with the construction of the critic match loss landscape, where all candidate weights are evaluated on the same reference batch taken from the final training episode. Under this fixed evaluation, earlier weights correspond to undertrained critics and therefore produce higher loss, whereas weights closer to the final solution yield lower loss. In this context, weights located in the initial region of the grid correspond to earlier, undertrained stages of learning and therefore produce higher loss values. In contrast, weights in the later region represent well-trained stages that contribute to convergence, resulting in lower loss. Since the final episode is itself a converged simulation, the combination of well-trained weights and stable system response yields a low loss, which appears as a darker region in the landscape plot.

This alignment between the converged behavior and the corresponding low-loss region in the landscape demonstrates that the loss landscape captures the optimization trend, even if it does not reflect the exact loss values at each training step. When training is unstable, the landscape highlights directions where the cost approximation is sensitive or poor. The value of the critic match loss landscape therefore lies not in reproducing the exact loss values during training, but in revealing the geometric characteristics of the critic parameter space and the qualitative behaviour of the optimization process.

2.4.4. Critic match loss landscape of final policy using random direction dim-reduction

To examine whether the observed loss landscape characteristics are intrinsic to the optimization process rather than artefacts of a specific projection, an alternative dimensionality reduction is considered. While PCA provides a linear projection that emphasizes directions of large weight variation, it does not uniquely determine the geometry of the loss surface. Therefore, a pair of random orthogonal directions is used as a complementary linear projection to assess the robustness of the landscape interpretation. It has to be mentioned that, only linear projections are considered here. Because the proposed loss landscape is constructed by explicitly reconstructing the critic network

parameters on a two-dimensional subspace and re-evaluating the critic match loss. This requires a linear projection that admits an explicit inverse mapping from the reduced coordinates to the original parameter space. As a result, nonlinear embedding methods such as t-SNE (t-distributed stochastic neighbor embedding) therefore isn't considered, since they are less suitable than linear dim-reduction methods for the reconstruction of valid network parameters for loss evaluation.



(a) 3-D loss of cart-pole ADHDP control

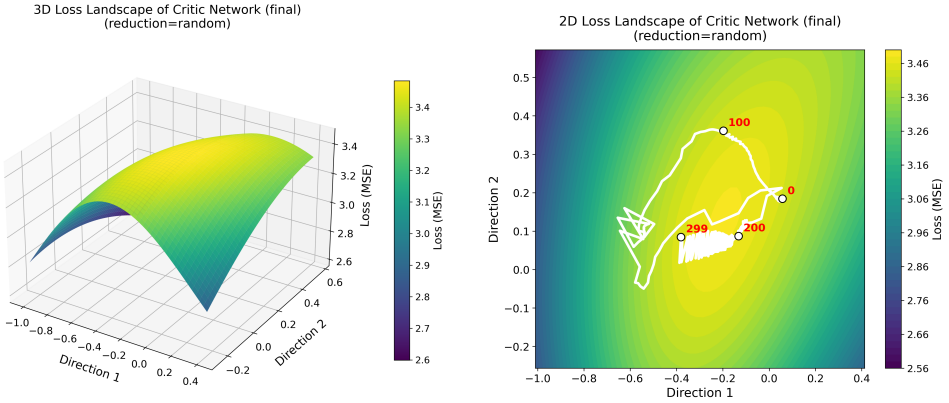
(b) 2-D loss path of cart-pole ADHDP control

Figure 2.10: 3-D and 2-D loss landscape with random direction dim-reduction of cart-pole ADHDP control

Figure 2.10a and Figure 2.10b show the 3-D surface and 2-D path when the weight space is projected onto two random orthogonal directions, instead of the PCA plane in Figure 2.8a and Figure 2.8b. Compared with the PCA projection, the absolute loss range on the grid is compressed and the slope becomes less steep. Nevertheless, the overall terrain remains a single tilted surface. The 2-D optimization path still progresses almost monotonically along a dominant descent direction, with only minor turns near dense contour regions as it approaches the minimum.

This indicates that, although the numerical scale and apparent steepness depend on the choice of projection, the essential optimization characteristics remain unchanged. In particular, a clear descent channel is preserved on the projected plane under the final-policy reference, which is consistent with stable weight evolution and a low system performance index $\tilde{J}_H$.

For spacecraft attitude control using ADHDP, Figure 2.11a and Figure 2.11b show the 3-D surface and the 2-D path when the weight space is projected onto a pair of random orthogonal directions. As expected, the absolute loss range and the detailed surface morphology vary with the projection. Under the random projection, the surface appears as a shallow arch rather than a well-defined basin. Along a ring near the top of this arch, the loss exhibits only weak variation, and the optimization path circulates around this region before



(a) 3-D loss landscape

(b) 2-D loss landscape

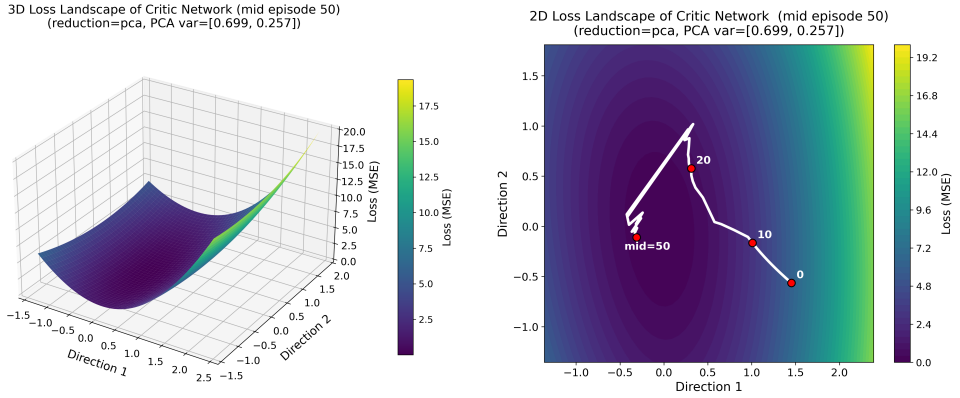
Figure 2.11: 3-D and 2-D loss landscape with random direction dim-reduction of spacecraft attitude ADHDP control

settling, as shown in Figure 2.11b. This results in a visible loop on the 2-D contour, reflecting a small effective gradient in tangential directions and a lack of a strong descent direction.

From the case of the spacecraft attitude control, the interpretation of the algorithm behavior does not depend on using PCA or random orthogonal directions for dim-reduction. Both projections reveal a difficult and skewed landscape with limited descent, which is consistent with unstable training and poor control performance.

2.4.5. Critic match loss landscape during training

A full movie of the online evolution is not feasible, since both the reference data, states and TD targets, and the critic parameters change at every step, making the objective itself move continuously. While the two-dimensional optimization path provides a compact visualization of how the critic weights move during training, a loss landscape constructed only under the final policy cannot reflect the geometry perceived by the critic at earlier stages of learning. To partially capture this temporal aspect, we fix the PCA plane built from the full sequence of episode-end critic weights and take mid-training snapshots. From the simulation setting in section 2.3, the mid-episode for cart-pole system is 50 and 150 for spacecraft attitude system. For each snapshot, the loss grid is re-centered at the episode-end weight of that episode. From Equation 2.1, this re-centering means that the coordinate of each grid is calculated based on the distance between that grid and the selected episode-end weight. The critic match loss is evaluated on that plane using the states and TD targets from that same episode. This yields a mid-training landscape that is directly comparable with the final-policy landscape. The mid-training surface as shown



(a) 3-D loss of cart-pole ADHDP control during training

(b) 2-D loss path of cart-pole ADHDP control during training

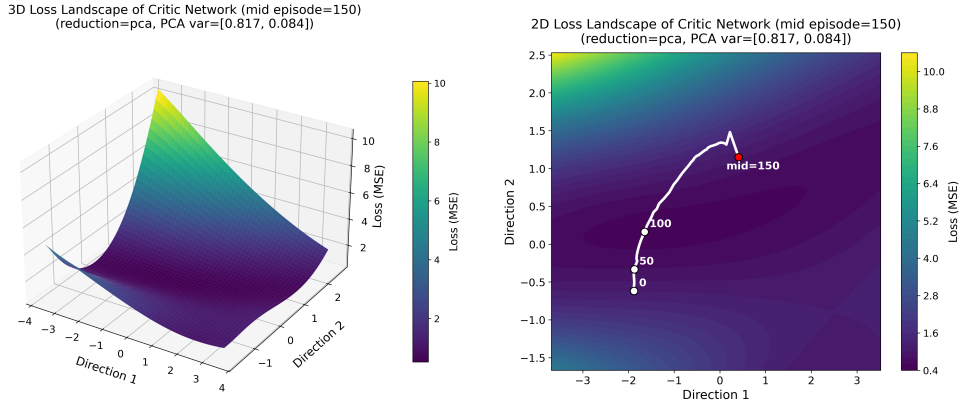
Figure 2.12: 3-D and 2-D loss landscape cart-pole ADHDP control during training

in Figure 2.12a and Figure 2.12b already exhibits a coherent and nearly convex basin aligned with the descent direction. The two-dimensional optimization path progresses steadily toward the basin center, with only minor adjustments near regions of denser contours. Compared with the final-policy landscape shown in Figure 2.8a, the basin at episode 50 is shallower and slightly broader, indicating that the local curvature is still developing.

Nevertheless, the overall geometry is characterized by a single dominant slope, weak anisotropy, and the absence of competing valleys. This consistency between the mid-training and final landscapes explains the stable and monotonic convergence observed during training.

The mid-training loss surface, shown in Figure 2.13a and Figure 2.13b, differs qualitatively from that of the cart-pole task. Around episode 150, the surface forms a shallow bowl-shaped region intersected by a ridge. Along this ridge, the two-dimensional optimization path turns and partially backtracks, before drifting through directions of low curvature. Compared with the final-policy landscape in Figure 2.9a, the mid-training surface is less cohesive, exhibiting weaker tangential curvature, while a narrow passage across the ridge remains visible. This shows that the critic optimization geometry is still evolving at this stage of training.

This behavior suggests a moving-target effect during learning. The descent directions observed in the mid-training landscape do not align with the low-loss region in the final landscape. It brings step-size sensitivity and oscillatory motion near the ridge. As a consequence, system performance remains unstable and the normalized cost stays large. Overall, the temporal loss landscapes suggest that the instability arises not only from non-convexity, but also from the evolving learning signals provided by the critic. As these signals change over time, policy updates repeat traveling across low-cost regions with



(a) 3-D loss of spacecraft attitude ADHDP control during training

(b) 2-D loss path of spacecraft attitude ADHDP control during training

Figure 2.13: 3-D and 2-D loss landscape spacecraft attitude ADHDP control during training

unstable features, preventing steady progress toward stable control.

2.5. Conclusion

The critic loss landscape visualization method for online reinforcement learning is proposed to analyze the training behavior of actor–critic control algorithms. By constructing a critic match loss surface on a low-dimensional plane and overlaying the optimization path, the method provides an interpretable representation of how critic parameters evolve during critic learning. To analyze the resulting landscapes beyond visual inspection, quantitative indices and a normalized system performance index are introduced, allowing the geometric properties of the landscape to be related to control outcomes. The method is evaluated using the Action-Dependent Heuristic Dynamic Programming (ADHDP) algorithm on the cart-pole and spacecraft attitude control problems. Through comparisons across projection methods and training stages, the simulations reveal distinct landscape characteristics associated with stable convergence and unstable learning. Overall, the results show that the proposed visualization and analysis framework enables both qualitative and quantitative interpretation of critic optimization behavior in online reinforcement learning, offering a practical tool for interpreting and comparing actor–critic algorithms in system control applications.

Appendix A: ADHDP algorithm details

The critic network and actor network are approximated using a MLP structure with one hidden layer.

In the critic network, the output $J(t)$ is in the form of Equation A.1. $w_{c_i}^{(2)}$ is the weight from the hidden layer to the output layer.

$$J(t) = \sum_{i=1}^{N_{hc}} w_{c_i}^{(2)}(t) p_i(t), \tag{A.1}$$

$$p_i(t) = \frac{1 - e^{-q_i(t)}}{1 + e^{-q_i(t)}}, i = 1, \cdots, N_{hc}, \tag{A.2}$$

$$q_i(t) = \sum_{j=1}^{n+1} w_{cij}^{(1)}(t) x_j(t), i = 1, \cdots, N_{hc}. \tag{A.3}$$

In the Equation A.2 and Equation A.3, q_i is the i^{th} hidden node input of the critic network, $w_{c_i}^{(1)}$ is the weight from the input layer to the hidden layer and p_i is the corresponding output of the hidden node, n is the dimension of the state vector and the additional input $x_{n+1} = 1$ represents the bias term.. N_{hc} is the total number of hidden nodes in the critic network. From Equation A.2, it can be seen that the hyperbolic-tangent function is used as the activation function at the hidden layer to add some nonlinearity to the training process. The linear function is used as the activation function at the output layer, as an example, as can be seen from Equation A.1. The choices for the activation function are decided according to the specific cases and users' experience.

Using a feed-forward network, the adaption in the action network is similar to the one in the critic network, while the inputs are the measured states, indicated with $x(t)$ in Figure 2.1, and the output is the action u. The associated equations for the action network are:

$$u(t) = \frac{1 - e^{-v(t)}}{1 + e^{-v(t)}}, \tag{A.4}$$

$$v(t) = \sum_{i=1}^{N_{ha}} w_{a_i}^{(2)}(t) g_i(t), \tag{A.5}$$

$$h_i(t) = \sum_{j=1}^{n} w_{a_{ij}}^{(1)}(t) x_j(t), \quad i = 1, \cdots, N_{ha}, \tag{A.6}$$

$$g_i(t) = \frac{1 - e^{-h_i(t)}}{1 + e^{-h_i(t)}}, i = 1, \cdots, N_{ha}, \tag{A.7}$$

where $v(t)$ is the input to the action node, and g_i and h_i are the output and the input of the hidden nodes of the action network, respectively. From Equation A.4, it can be seen that the hyperbolic-tangent function is used as the activation function at the hidden layer. The hyperbolic tangent function is also used as the activation function at the output layer, as can be seen from Equation A.7. Similar to the critic network, the choices for the activation function are decided according to the specific cases and users' experiences. The training of the two neural networks is based on backpropagation. With the chain rule, the adaption of the critic network is summarized as follows. From hidden to the output layer

$$\Delta w_{c_ij}^{(2)}(t) = l_c(t)\left[-\frac{\partial E_c(t)}{\partial w_{c_ij}^{(2)}(t)}\right], \tag{A.8}$$

$$\frac{\partial E_c(t)}{\partial w_{c_ij}^{(2)}(t)} = \frac{\partial E_c(t)}{\partial J(t)}\frac{\partial J(t)}{\partial w_{c_ij}^{(2)}(t)} = \gamma e_c(t) p_i(t), \tag{A.9}$$

where $\Delta w_{c_i}^{(2)}(t)$ indicates the weight from the hidden to the output layer of the critic network and $E_c(t)$ denotes the instantaneous critic error function.

From the input to the hidden layer

$$\Delta w_{c_ij}^{(1)}(t) = l_c(t)\left[-\frac{\partial E_c(t)}{\partial w_{c_ij}^{(1)}(t)}\right], \tag{A.10}$$

$$\begin{aligned}\frac{\partial E_c(t)}{\partial w_{c_ij}^{(1)}(t)} &= \frac{\partial E_c(t)}{\partial J(t)}\frac{\partial J(t)}{\partial p_i(t)}\frac{\partial p_i(t)}{\partial q_i(t)}\frac{\partial q_i(t)}{\partial w_{c_ij}^{(1)}(t)} \\ &= \gamma e_c(t) w_{c_ij}^{(2)}(t)\left[\frac{1}{2}\left(1 - p_i^2(t)\right)\right] x_j(t),\end{aligned} \tag{A.11}$$

where $\Delta w_{c_i}^{(1)}(t)$ indicates the weight from the input to the hidden layer of the critic network.

The update rule for the nonlinear multi-layer action network also contains two sets of equations.

From the hidden to the output layer

$$\Delta w_{a_i}^{(2)}(t) = l_a(t)\left[-\frac{\partial E_a(t)}{\partial w_{a_i}^{(2)}(t)}\right], \tag{A.12}$$

$$\begin{aligned}\frac{\partial E_a(t)}{\partial w_{a_i}^{(2)}(t)} &= \frac{\partial E_a(t)}{\partial J(t)}\frac{\partial J(t)}{\partial u(t)}\frac{\partial u(t)}{\partial v(t)}\frac{\partial v(t)}{\partial w_{a_i}^{(2)}(t)} \\ &= e_a(t)\sum_{i=1}^{N_{hc}}\left[w_{c_i}^{(2)}(t)\frac{1}{2}\left(1 - p_i^2(t)\right) w_{c_{i,n+1}}^{(1)}(t)\right]\left[\frac{1}{2}\left(1 - u^2(t)\right)\right] g_i(t),\end{aligned} \tag{A.13}$$

where $\Delta w_{a_i}^{(2)}$indicates the weight from the hidden to the output layer of the actor network and $E_a(t)$ denotes the instantaneous error function of the actor network.

In Equation A.13, $\partial J(t)/\partial u(t)$ is obtained from the critic network by changing variables and by the chain rule. The term $w_{c_{i,n+1}}^{(1)}$ is the weight associated with the input element from the action network output. Here, i is the i^{th} node of the input layer, and n is the number of nodes of the input layer.

From the input to the hidden layer

$$\Delta w_{a_{ij}}^{(1)}(t) = l_a(t)\left[-\frac{\partial E_a(t)}{\partial w_{a_{ij}}^{(1)}(t)}\right], \tag{A.14}$$

$$\begin{aligned}\frac{\partial E_a(t)}{\partial w_{a_{ij}}^{(1)}(t)} &= \frac{\partial E_a(t)}{\partial J(t)}\frac{\partial J(t)}{\partial u(t)}\frac{\partial u(t)}{\partial v(t)}\frac{\partial v(t)}{\partial g_i(t)}\frac{\partial g_i(t)}{\partial h_i(t)}\frac{\partial h_i(t)}{\partial w_{a_{ij}}^{(1)}(t)} \\ &= e_a(t)\sum_{i=1}^{N_{hc}}\left[w_{c_i}^{(2)}(t)\frac{1}{2}\left(1-p_i^2(t)\right)w_{c_{i,n+1}}^{(1)}(t)\right] \\ &\left[\frac{1}{2}\left(1-u^2(t)\right)\right]w_{a_i}^{(2)}(t)\left[\frac{1}{2}\left(1-g_i^2(t)\right)\right]x_j(t),\end{aligned} \tag{A.15}$$

where $w_{a_{ij}}^{(1)}(t)$ indicates the weight from the input to the hidden layer of the actor network.

In ADHDP implementations, Equation A.9 and Equation A.11 are used to update the weights in the critic network, while Equation A.13 and Equation A.15 are used to update the weights in the action network.

References

[1] J. Ibarz, J. Tan, C. Finn, M. Kalakrishnan, P. Pastor, and S. Levine. "How to train your robot with deep reinforcement learning: lessons we have learned". In: *The International Journal of Robotics Research* 40.4-5 (2021), pp. 698–721.

[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.* "Mastering the game of go without human knowledge". In: *nature* 550.7676 (2017), pp. 354–359.

[3] S. Silvestrini and M. Lavagna. "Deep Learning and Artificial Neural Networks for Spacecraft Dynamics, Navigation and Control". In: *Drones* 6.10 (2022). issn: 2504-446X. doi: 10.3390/drones6100270. url: https://www.mdpi.com/2504-446X/6/10/270.

[4] D. Lee, H. Seo, and M. W. Jung. "Neural basis of reinforcement learning and decision making". In: *Annual review of neuroscience* 35.1 (2012), pp. 287–308.

[5] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. "Benchmarking deep reinforcement learning for continuous control". In: *International conference on machine learning*. PMLR. 2016, pp. 1329–1338.

[6] C. E. Oestreich, R. Linares, and R. Gondhalekar. "Autonomous six-degree-of-freedom spacecraft docking with rotating targets via reinforcement learning". In: *Journal of Aerospace Information Systems* 18.7 (2021), pp. 417–428.

[7] M. Tipaldi, R. Iervolino, and P. R. Massenio. "Reinforcement learning in spacecraft control applications: Advances, prospects, and challenges". In: *Annual Reviews in Control* 54 (2022), pp. 1–23.

[8] S. Rafiee, M. Kankashvar, P. Mohammadi, and H. Bolandi. "Active fault-tolerant attitude control based on Q-learning for rigid spacecraft with actuator faults". In: *Advances in Space Research* 74.3 (2024), pp. 1261–1275.

[9] C. Wei, J. Luo, H. Dai, Z. Bian, and J. Yuan. "Learning-based adaptive prescribed performance control of postcapture space robot-target combination without inertia identifications". In: *Acta Astronautica* 146 (2018), pp. 228–242.

[10] H. Gao. "Attitude Stabilization Control for Combined Spacecraft". Available at https://kns.cnki.net/KCMS/detail/detail.aspx?dbname=CDFDLAST2021&filename=1020401662.nh. PhD thesis. Harbin Institute of Technology, 2019.

[11] C. Packer, K. Gao, J. Kos, P. Krähenbühl, V. Koltun, and D. Song. "Assessing generalization in deep reinforcement learning". In: *arXiv preprint* (2018). arXiv:1810.12282.

[12] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. “Sim-to-real transfer of robotic control with dynamics randomization”. In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 3803–3810.

[13] X. Liang, D. Bao, and S. S. Ge. “Modeling of Neuro-Fuzzy System With Optimization Algorithm as a Support in System Boundary Capability Online Assessment”. In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 70.8 (2023), pp. 2974–2978. doi: 10.1109/TCSII.2023.3255419.

[14] Y. Zhang, X. Liang, D. Li, S. S. Ge, B. Gao, H. Chen, and T. H. Lee. “Adaptive Safe Reinforcement Learning With Full-State Constraints and Constrained Adaptation for Autonomous Vehicles”. In: *IEEE Transactions on Cybernetics* 54.3 (2024), pp. 1907–1920. doi: 10.1109/TCYB.2023.3283771.

[15] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein. “Visualizing the loss landscape of neural nets”. In: *Advances in neural information processing systems* 31 (2018).

[16] R. Sullivan, J. K. Terry, B. Black, and J. P. Dickerson. “Cliff diving: Exploring reward surfaces in reinforcement learning environments”. In: *arXiv preprint arXiv:2205.07015* (2022).

[17] N. Rahn, P. D’Oro, H. Wiltzer, P.-L. Bacon, and M. Bellemare. “Policy optimization in a noisy neighborhood: On return landscapes in continuous control”. In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 30618–30640.

[18] R. Y. Bekci and M. Gümüş. “Visualizing the loss landscape of actor critic methods with applications in inventory optimization”. In: *arXiv preprint arXiv:2009.02391* (2020).

[19] J. Schneider, P. Schumacher, D. Häufle, B. Schölkopf, and D. Büchler. “Investigating the impact of action representations in policy gradient algorithms”. In: *arXiv preprint arXiv:2309.06921* (2023).

[20] J. Si, A. G. Barto, W. B. Powell, and D. Wunsch. *Handbook of learning and approximate dynamic programming*. Vol. 2. John Wiley & Sons, 2004.

[21] A. G. Barto, R. S. Sutton, and C. W. Anderson. “Neuronlike adaptive elements that can solve difficult learning control problems”. In: *IEEE transactions on systems, man, and cybernetics* 5 (1983), pp. 834–846.

[22] P. Huang, Y. Lu, M. Wang, Z. Meng, Y. Zhang, and F. Zhang. “Postcapture attitude takeover control of a partially failed spacecraft with parametric uncertainties”. In: *IEEE Transactions on Automation Science and Engineering* 16.2 (2018), pp. 919–930.

3

Adapting Critic Match Loss Landscape Visualization to Off-policy Reinforcement Learning

This work extends an established critic match loss landscape visualization method from online to off-policy reinforcement learning (RL), aiming to reveal the optimization geometry behind critic learning. Off-policy RL differs from stepwise online actor-critic learning in its replay-based data flow and target computation. Based on these two structural differences, the critic match loss landscape visualization method is adapted to the Soft Actor-Critic (SAC) algorithm by aligning the loss evaluation with its batch-based data flow and target computation, using a fixed replay batch and precomputed critic targets from the selected policy. Critic parameters recorded during training are projected onto a principal component plane, where the critic match loss is evaluated to form a 3-D landscape with an overlaid 2-D optimization path. Applied to a spacecraft attitude control problem, the resulting landscapes are analyzed both qualitatively and quantitatively using sharpness, basin area, and local anisotropy metrics, together with temporal landscape snapshots. Comparisons between convergent SAC, divergent SAC, and divergent Action-Dependent Heuristic Dynamic Programming (ADHDP) cases reveal distinct geometric patterns and optimization behaviors under different algorithmic structures. The results demonstrate that the adapted critic match loss visualization framework serves as a geometric diagnostic tool for analyzing critic optimization dynamics in replay-based off-policy RL-based control problems.

This chapter is based on the revised manuscript: J. Liu, J. Guo, and E. Gill, "Adapting Critic Match Loss Landscape Visualization to Off-policy Reinforcement Learning," submitted to *Astrodynamics*, DOI: https://doi.org/10.48550/arXiv.2603.14589.

3.1. Introduction

Reinforcement learning has been a hot spot for research in recent years. While the research was initiated by the computer science community, it quickly spread its applications and related theory development also to other research communities.

With its wide applications in optimization [1], control [2], and planning [3], reinforcement learning has shown potential in solving decision-making problems. However, reinforcement learning algorithms often suffer from limited generalization capability, resulting in unstable training performance and inconsistent control outcomes across different environments. As a result, researchers are concerned about the interpretation of the performance of the reinforcement learning algorithms, rather than only treating it as a black box [4]. Interpretation methods, such as decision trees [5], natural language explanations [6], and visualization methods [7], have been developed and used to interpret the algorithm's behavior in decision making, planning, and other aspects. Among them, the visualization method is a widely used method for its intuitiveness, readability, and ease of interpretation by readers. The loss landscape is an index that is often chosen by researchers to visualize the internal process during the training of reinforcement learning methods [8], and also the characteristics of the neural networks that are used for approximation in the learning methods [9].

The authors previously developed a method for visualizing the critic-matching loss landscape to analyze actor–critic learning behavior. That work focused on online reinforcement learning and demonstrated how the visualization reflects critic optimization patterns under continuously updated state–action data. Unlike online algorithms, which interact with the environment and update parameters in a stepwise manner, off-policy reinforcement learning adopts a replay-based training regime with decoupled data usage and parameter updates. Although the agent continues to interact with the environment, critic learning is performed on mini-batches sampled from a replay buffer, and target values are computed using delayed or target networks rather than being directly tied to the most recent interaction. Because of these structural differences, the method cannot be directly applied to off-policy RL without further adaptation. Bridging this gap ensures that the visualized loss geometry remains meaningful under the batch-based learning characteristic of off-policy RL.

The present work adapts the critic match loss landscape visualization method to suit the off-policy RL setting, using the Soft Actor–Critic (SAC) algorithm [10] as a representative case. SAC is chosen due to its strong and stable performance in control tasks [11, 12]. To accommodate the replay-based training structure of SAC, the visualization method is adapted in two main aspects. First, to address the data flow difference, a fixed batch of replay data is used instead of continuously collected online samples, and the critic weight recording frequency is defined at the scale of neural network updates rather than episode boundaries. Second, to adapt the critic match loss

computation, the target values are precomputed and fixed based on the final policy, incorporating the twin-critic structure and entropy regularization of SAC. With these adaptations, critic parameters recorded during training are projected onto principal directions identified through principal component analysis (PCA), and the critic match loss is evaluated to construct a three-dimensional loss landscape with an overlaid two-dimensional optimization path. Applied to a spacecraft attitude control problem, the resulting loss landscapes are analyzed both qualitatively and quantitatively using sharpness, basin area, and local anisotropy metrics. Temporal landscape snapshots are further constructed to examine how critic geometry evolves during training and to identify potential instability events. The results show that, under the frozen-target surrogate objective, the proposed visualization method provides structural insight into critic optimization behavior in the off-policy RL setting, extending its applicability beyond online actor–critic methods. Comparisons among convergent SAC, divergent SAC, and Action-Dependent Heuristic Dynamic Programming (ADHDP) cases reveal how different algorithmic structures shape critic optimization geometry and its relation to control behavior.

This paper is organized as follows. Section 2 introduces the loss function visualization technique, outlines the critic match loss landscape method and describes the adaptations required for its use in off-policy reinforcement learning. The quantitative indices for analyzing loss landscapes are introduced. Section 3 shows the setting of the spacecraft attitude system used for demonstrating the control algorithm's critic match loss landscape. Section 4 presents the critic match loss landscape visualization results, including comparative analyses between SAC and ADHDP, temporal landscape snapshots, and a discussion of the interpretation scope of the method. Section 5 validates the proposed critic match loss landscape method using SAC control on the Cart–Pole benchmark as a case study. Section 6 concludes the paper.

3.2. Critic match loss landscape visualization method and its adaptation to off-policy RL

In this section, the visualization method for interpreting online RL algorithms is introduced. The Soft Actor Critic (SAC) algorithm is also provided, which serves as an object to be interpreted using the visualization method.

3.2.1. Visualizing the loss function

In reinforcement learning, neural networks are used to approximate functions, often with a large number of parameters. The loss landscape, which shows how the loss varies with these parameters, can reveal properties such as flatness, sharpness, local minima, and saddle points. However, the high dimensionality of the parameter space makes direct visualization difficult. To address this, the Contour Plots and Random Directions method is proposed [9]. The idea is to select two directions, $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$, and plot the loss over the 2-D plane they span.

The resulting surface is a projection slice of the full loss landscape. The loss value in this plane is computed as

$$f(\alpha,\beta) = \mathcal{L}(\boldsymbol{\zeta}^* + \alpha\,\boldsymbol{\delta} + \beta\,\boldsymbol{\eta})\,, \tag{3.1}$$

where $\boldsymbol{\zeta}^*$ is the chosen center point. α and β represent displacements along the two directions. $\mathcal{L}$ is the loss function. This reduces the multidimensional problem to a 3-D plot, with (α,β) on the horizontal plane and $f(\alpha,\beta)$ on the vertical axis, making the structure of the loss landscape explicit and easy to interpret.

3

3.2.2. Soft Actor-Critic (SAC) algorithm

In previous work, the loss visualization technique has been proposed to interpret the performance of online reinforcement learning algorithms with an actor-critic structure. To further extend the proposed visualization method beyond online learning, this study applies it to the Soft Actor–Critic (SAC) algorithm, which represents an off-policy reinforcement learning framework known for its stable control performance.

Figure 3.1 shows the structure of the SAC algorithm. Similar to the ADHDP algorithm, it also employs an actor and critic structure. The actor is approximated to generate the actions, and the critic network is used to approximate the cost function. Different from the ADHDP algorithm, it uses two critic networks. One critic updates its parameters with a delay relative to the other. This delayed update mechanism reduces overestimation bias in value approximation and stabilizes training, leading to a more accurate estimation of the cost function.

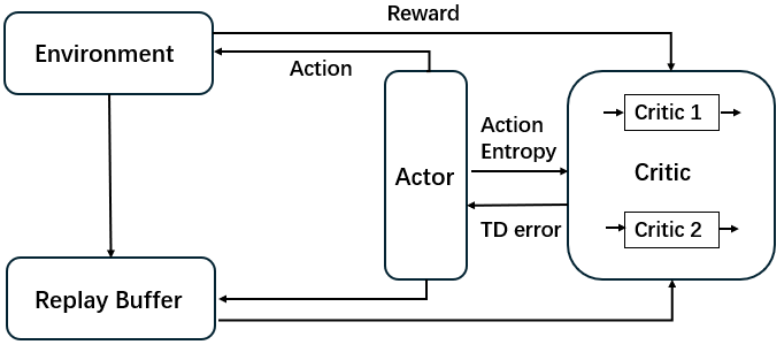


Figure 3.1: SAC structure diagram

Another factor where the SAC algorithm differs from the ADHDP algorithm is that the SAC algorithm is an off-policy reinforcement learning algorithm that

optimizes a stochastic policy using an entropy-augmented objective function, which is the cost function in the case of the ADHDP algorithm. The goal of the Soft Actor–Critic (SAC) algorithm is to maximize both the expected cumulative reward and the entropy of the policy:

$$J(\pi) = \sum_{t=0}^{\infty} \mathbb{E}_{(s_t,a_t)\sim d_\pi} \left[r(s_t, a_t) + \alpha_{\text{temp}} \mathcal{H}(\pi(\cdot|s_t)) \right], \tag{3.2}$$

where $\mathcal{H}(\pi(\cdot|s_t)) = -\mathbb{E}_{a_t\sim\pi}[\log \pi(a_t|s_t)]$ is the entropy term, and $\alpha_{\text{temp}} > 0$ is the temperature coefficient that balances the trade-off between reward maximization and exploration.

The critic networks are trained to minimize the Bellman residual using the target Q-value:

$$J_{\mathcal{Q}}(\mathbf{w}_c) = \mathbb{E}_{(s_t,a_t,r_t,s_{t+1})\sim\mathcal{D}} \left[\left(\mathcal{Q}_{\mathbf{w}_c}(s_t, a_t) - y_t^{\text{td}} \right)^2 \right], \tag{3.3}$$

where the target value y_t^{td} is computed as:

$$y_t^{\text{td}} = r_t + \gamma \, \mathbb{E}_{a_{t+1}\sim\pi} \left[\mathcal{Q}_{\bar{\mathbf{w}}_c}(s_{t+1}, a_{t+1}) - \alpha_{\text{temp}} \log \pi(a_{t+1}|s_{t+1}) \right]. \tag{3.4}$$

Here, s_t and a_t denote the system state and action at time step t, respectively, while r_t is the immediate reward. The term d_π represents the state–action distribution under policy π. The parameter $\mathbf{w}_c$ refers to the critic network weights, and $\bar{\mathbf{w}}_c$ denotes the slowly updated target critic network weights. The discount factor $\gamma \in (0, 1)$ determines the importance of future rewards. The replay buffer $\mathcal{D}$ stores sampled transitions (s_t, a_t, r_t, s_{t+1}) that are used to train the critic network.

3.2.3. Visualizing the critic match loss function for SAC algorithm

The idea of visualizing the critic match loss function was previously developed by the authors to analyze the critic network behavior of online reinforcement learning algorithm. In that context, the visualization was shown to reflect critic optimization patterns under continuously updated state–action data. Besides online RL algorithms, off-policy reinforcement learning algorithms are also widely used in control scenarios. However, off-policy RL algorithms often have more complicated structures. So it's important to see how the established critic match loss landscape visualization method can be adapted to off-policy RL settings. In this work, the Soft Actor Critic Algorithm is selected as the sample algorithm for off-policy RL.

Before introducing the adaptation of the critic match loss visualization method to off-policy reinforcement learning, the visualization procedure for online RL is briefly summarized. For an online actor–critic reinforcement learning algorithm, the critic network is employed to approximate the cost function, and its approximation accuracy directly affects the control performance. The temporal-difference (TD) error reflects this approximation accuracy and is iteratively minimized during critic training. However, both the TD error and the

distribution of system states evolve continuously as training progresses, which makes it difficult to represent the training process using a single, consistent TD-error surface. To address this issue, the critic match loss is introduced as a surrogate measure to represent the evolution of the training signal in a fixed reference setting.

To capture the training evolution, the critic weight vector is recorded at the end of each training episode, forming a sequence of weight snapshots that reflects the progression of critic updates over the entire training process. Since the critic weights are high-dimensional, dimensionality reduction is required for visualization. As indicated in Equation 3.1, two directions are selected to serve as the axes of the loss landscape. These directions are obtained by applying principal component analysis (PCA) to the set of recorded critic weight vectors $\mathbf{w}_c$, yielding two mutually orthogonal basis vectors. Denoted as $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$, these directions define a low-dimensional parameter plane for visualizing variations in $\mathbf{w}_c$. Rather than depicting the loss along a single dimension, the loss surface is explored in two dimensions by sampling the parameters α and β over a predefined grid. For each (α, β) pair, the critic weights are reconstructed as $\mathbf{w}_c' = \mathbf{w}_c + \alpha\boldsymbol{\delta} + \beta\boldsymbol{\eta}$, and the corresponding critic match loss is evaluated.

For each grid point, the loss is computed using fixed input–target pairs. The inputs consist of system states collected from the final training episode, while the targets are the corresponding temporal-difference targets computed at the final epoch. This procedure yields a three-dimensional loss landscape that characterizes the local loss geometry around the final policy. By projecting the loss landscape onto the same PCA plane, the recorded critic weight trajectory can be overlaid as a two-dimensional optimization path. This path provides a qualitative representation of how the critic parameters evolve relative to the underlying loss geometry during training. Although this construction does not reproduce the exact time-varying TD objective used throughout training, it offers an interpretable view of the critic's optimization behavior and supports the analysis of convergence and instability in online reinforcement learning.

To adapt the critic match loss landscape technique from online to off-policy setting, it's necessary to compare the structure differences between online RL and off-policy RL, which will show the aspects the match loss landscape technique has to be adapted.

1. Firstly, the data flow is different. Different from stepwise online actor-critic reinforcement learning, in which the sample data is collected per time step and the weight updates accordingly, SAC is an off-policy reinforcement learning algorithm with replay-based updates. Unlike strictly offline batch RL, the SAC implementation used in this work continues to interact with the environment throughout training. At each time step, the agent interacts with the environment and stores the resulting transition (s_t, a_t, r_t, s_{t+1}) into a replay buffer. Environment interaction continues throughout training and the replay buffer grows incrementally as new transitions are collected. Once the buffer contains enough samples to form a mini-batch, the neural networks are updated by repeatedly sampling batches from the buffer. And the weights of

the neural networks update accordingly.

2. Secondly, the target calculation in TD error is different. TD error is used to calculate the critic match loss and generate the local geometry of the critic match loss function, which can show how the critic match loss varies in the neighborhood of the final policy. From Equation 3.4, the TD error can be expressed as

$$\begin{aligned} e_t^{\mathrm{TD}} = & Q_{\mathbf{w}_c}(s_t, a_t) \\ & - \Big(r_t + \gamma \min_{i=1,2} Q_{\bar{\mathbf{w}}_{ci}}(s_{t+1}, a_{t+1}) \\ & - \alpha_{\mathrm{temp}} \log \pi(a_{t+1} \mid s_{t+1}) \Big) \end{aligned}, \tag{3.5}$$

where $Q_{\mathbf{w}_c}(s_t, a_t)$ is the Q value of the current critic network, $Q_{\bar{\mathbf{w}}_{ci}}$ is the current target network which uses soft update, r_t is the current reward, γ is the discount factor, $\pi(a_{t+1} \mid s_{t+1})$ is the probability density of action a_{t+1} under the state s_{t+1}, α_{temp} is the temperature coefficient that controls the trade-off between policy exploration and exploitation and $a_{t+1} \sim \pi(\cdot \mid s_{t+1})$ is the next action sampled under the current policy with state s_{t+1}. For online RL, TD error expression is much more simple, which is

$$e_c(t) = [r(t) + \gamma J(t)] - J(t-1). \tag{3.6}$$

The difference in the target computation in TD error will determine the calculation of critic match loss.

From the discussion above, it can be seen that the structural differences between online and off-policy reinforcement learning mainly lie in (i) the data usage flow and (ii) the target computation mechanism for critic updates. Accordingly, the visualization method is adapted along two directions: adaptation to the data flow and adaptation to the critic match loss computation.

A. Adaptation to the off-policy data flow

To adapt to the data flow in off-policy RL, two design choices are introduced.

(1) Using a fixed batch of replay data instead of continuously collected online samples. In online RL, the sample distribution evolves as the policy changes and new trajectories are generated. For online RL, the states and loss from the last training episode are used for the critic match loss of each grid. Using the data from the last training episode, the final policy is evaluated through the critic loss landscape. In off-policy RL, critic training is batch-based and driven by replay samples. Therefore, a fixed batch is sampled from the replay buffer, which is consistent with SAC's training mechanism.

(2) Recording the critic weights at a frequency defined by network update steps. For an online reinforcement learning method, the weight recording frequency is selected based on the real-world dynamics. For the online RL visualization, the weight is recorded when each episode ends, which either meets the simulation time limit or system states limit. However, as described above, the data usage flow for neural networks and the weight update flow is different in the case of an off-policy reinforcement learning algorithm. SAC updates the networks by repeatedly sampling mini-batches from the replay

buffer, and a large number of parameter updates may occur within a single episode. As a result, the weight recording frequency is chosen on the same scale as the neural network update steps. Specifically, the network parameters are recorded at fixed intervals during training. These recorded parameter sets are then used to project the loss landscape over the entire training process.

B. Adaptation to the critic match loss computation in SAC

A fixed batch of state "s_t, a_t, r_t, s_{t+1}" samples are collected to calculate the corresponding Bellman targets y_t^{td} using Equation 3.4, where the expectation term over the next action is evaluated using the final policy. To adapt to this critic match loss computation scheme, three design choices are introduced.

(1) SAC Bellman targets are precomputed and fixed for a selected batch of replay data. In SAC, the target value depends on the twin-critic structure, the entropy term, and policy sampling. During visualization, the critic parameters are systematically perturbed on a weight grid that does not correspond to actual training states. Therefore, recomputing targets for each grid point would introduce ambiguity and make the loss definition inconsistent. Fixing the targets computed from the final policy provides a well-defined reference for evaluating the critic loss across the weight grid.

(2) The visualization focuses on the primary critic network within the twin-critic architecture. Both critics are trained using the same loss formulation and share an identical target definition. Therefore, focusing the visualization on a single critic network does not alter the SAC training objective and allows the loss landscape to be defined consistently.

(3) The actor network and the temperature parameter are kept fixed during visualization. Since the goal is to examine how the critic loss varies with respect to critic parameters alone, allowing the actor or entropy coefficient to change would couple multiple sources of variation. Freezing these components isolates the local loss geometry of the critic while remaining consistent with the final stage of SAC training.

From the method description above, there are two main adaptions when the critic match loss visualization is applied to off-policy actor-critic RL algorithm. The first main adaption is caused by the data flow difference between online and off-policy RL. For the adaption in data flow, two design choices are made (1) using a fixed batch of replay data instead of continuously collected online samples; (2) critic weight recording frequency. For the adaption in calculation critic match loss, three design choices are made. (1) precomputing and fixing the target values that combine the twin-critic structure and the entropy term; (2) focusing the analysis on the primary critic within the twin-critic pair; and (3) freezing the actor and temperature parameters to isolate the critic's local loss geometry. These modifications align the visualization method with the SAC training mechanism while preserving its original interpretive purpose, enabling consistent comparison between online and off-policy RL paradigms.

The above description of constructing critic match loss landscape is illustrated using the final trained policy. Yet the formulation is not restricted to the terminal stage of training. By recording the critic parameters, associated

target quantities, and a representative replay batch at any selected time point, and freezing these components, a locally stationary surrogate objective can be defined for that stage. Training then proceeds with further environment interaction and buffer growth. The resulting landscape therefore characterizes the critic geometry corresponding to a specific policy snapshot. By constructing such frozen-target landscapes at multiple training stages, the method enables a temporal analysis of critic optimization behavior. While each surface remains a static approximation, the sequence of snapshots provides insight into how critic stability and geometric structure evolve throughout learning.

By projecting the 3-D landscape onto the 2-D PCA plane obtained from the recorded critic weights, the optimization trajectory of the critic parameters during SAC training can be visualized on the same contour map. This 2-D path qualitatively represents how the weights evolve in relation to the underlying loss geometry, thereby revealing the optimization trend and convergence behavior throughout the training process.

This procedure constructs a stationary surrogate of the critic's training objective under a selected policy and a qualitative representation of the projected optimization path. Although it does not reproduce the exact stepwise temporal difference objectives used during SAC training, it enables the visualization of how the critic match loss varies in the final policy, providing a clear depiction of the local geometry that characterizes the optimization behavior of the critic of SAC.

3.2.4. Quantitative analysis of loss landscape

The critic match loss landscape is derived using the method in subsection 3.2.3. It will directly show the qualitative representation of the optimization path. To further enable objective comparison and interpretation beyond visual inspection, a quantitative analysis of the loss landscape is introduced here.

Three indices are considered in this work: *sharpness*, *basin area*, and *local anisotropy*. These indices capture different aspects of the loss geometry around the final critic parameters and provide complementary descriptions of the training outcome.

Since the magnitude of the critic match loss varies significantly across algorithms and control systems, a direct comparison of raw loss values is not meaningful. Therefore, the loss landscape is first expressed relative to the final critic parameters. Let $L(\alpha, \beta)$ denote the critic match loss on the PCA plane, and let (α^*, β^*) be the final critic parameters with loss value L^*. A relative loss surface is defined as

$$\Delta L(\alpha, \beta) = L(\alpha, \beta) - L^*. \tag{3.7}$$

To remove scale dependence, ΔL is normalized by a robust internal scale, chosen as the inter quartile range (IQR) of ΔL over the grid. The resulting dimensionless surface $\tilde{L}$ enables consistent geometric interpretation across different training runs.

Sharpness Sharpness describes how rapidly the loss increases when the critic parameters are perturbed away from the final point. For a given radius ϵ, it is computed as

$$\mathrm{Sharp}_\epsilon = \max_{\partial_{\mathrm{dir}} \in [0,2\pi)} \tilde{L}(\alpha + \epsilon \cos \partial_{\mathrm{dir}},\ \beta + \epsilon \sin \partial_{\mathrm{dir}})\,. \tag{3.8}$$

This index represents the worst-case local sensitivity of the critic loss within a neighborhood of radius ϵ. ∂_{dir} denotes the angular direction on the circle of radius ϵ and it is unrelated to the network parameter vector.

From the perspective of reinforcement learning evolution, a larger sharpness indicates that the final critic parameters lie in a region where deviations are strongly penalized, suggesting a locally well-defined solution with a clear descent structure. In contrast, a small sharpness value corresponds to a flat or weakly constrained region, which is often associated with unstable or non-convergent training behavior.

Basin area The basin area characterizes the spatial extent of low-loss regions surrounding the final critic parameters. It is defined as the area of the region where the normalized loss remains below a prescribed threshold ρ,

$$A_\rho = \mathrm{Area}\left\{(\alpha, \beta) \mid \tilde{L}(\alpha, \beta) \le \rho\right\}. \tag{3.9}$$

This quantity provides a measure of how broadly the critic can vary while maintaining a comparable loss level.

In the context of reinforcement learning, a larger basin area suggests that the training process has converged to a solution that is robust to parameter perturbations, whereas a very small basin indicates a fragile solution that depends on fine parameter tuning. It should be noted that when training fails to converge, the loss landscape may not exhibit a closed basin structure, in which case this value does not convey meaningful information.

Local anisotropy Local anisotropy describes the directional imbalance of the loss landscape near the final critic parameters. A quadratic approximation of $\tilde{L}$ is constructed in a small neighborhood of (α^*, β^*), yielding a 2×2 Hessian matrix H on the PCA plane. The degree of anisotropy is measured by the logarithm of the condition number,

$$\log \kappa = \log\left(\frac{\lambda_{\max}(H)}{\lambda_{\min}(H)}\right), \tag{3.10}$$

where $\lambda_{\max}$ and $\lambda_{\min}$ are the largest and smallest eigenvalues of H, respectively.The condition number of the resulting Hessian matrix on the PCA plane captures the relative difference between steep and flat directions of the loss.

From an optimization viewpoint, strong anisotropy indicates an ill-conditioned valley, where progress is sensitive to step size and update direction. In reinforcement learning, such landscapes are often associated with slow

convergence or instability, whereas more isotropic curvature corresponds to smoother and more reliable training dynamics.

Together, these indices provide a compact quantitative description of the critic loss landscape. When combined with the visualization of the optimization path, they offer insight into how different reinforcement learning algorithms evolve in parameter space and why their training outcomes may differ.

3.3. Spacecraft attitude control setting

In this section, the spacecraft attitude system is selected as the sample system for SAC control. The control results of using the SAC algorithm are shown.

In this work, the combined spacecraft in the ADR scenario will be used as the control object for the SAC algorithm. From [13], the combined spacecraft in an ADR mission is a system affected by different kinds of uncertainty. In this work, to test the control algorithm for the post-capture stage in a gradual way, the simplest case is studied first, where the only uncertainty comes from the inertial parameters. The following assumptions are made:

- The target is firmly grasped by rigid robotic manipulators mounted on a rigid servicing spacecraft.
- After capture, all manipulator joints are locked [14].
- The target is rigid, uncooperative, and has no control capability.

With these conditions, the combined spacecraft is treated as a single rigid body with unknown inertial parameters. The dynamic model of the spacecraft attitude system is described in detail in Appendix A.

In this study, the reinforcement learning algorithm is trained in simulation rather than directly on the physical spacecraft, yielding an initial stabilizing control policy represented by the actor network. During operation, the learned policy is deployed as a feedback controller that generates control inputs based on the observed spacecraft states. Pre-training in simulation improves safety and stability by avoiding the need to start from an untrained policy on the real system. In this training stage, the critic match loss landscape can also be used to analyze the evolution of the critic optimization and the associated control behavior, providing insight into the learning process before deployment.

During training, the RL agent interacts with the spacecraft dynamics simulation module to generate state–action–transition samples. These transitions are stored in a replay buffer and reused for mini-batch updates following the off-policy training scheme of SAC. Therefore, the training process does not assume prior knowledge of the spacecraft dynamics. The critic and actor networks are learned directly from interaction data.

Unless otherwise specified, the general simulation settings for the spacecraft attitude control experiments are summarized below. The spacecraft attitude control policy was trained using the Soft Actor–Critic (SAC) algorithm from the Stable-Baselines3 library [15]. The simulation environment was the

quaternion-based attitude dynamics model described in Appendix A. The SAC agent used the default multi-layer perceptron (MLP) policy network. The training was run for a total of 2×10^5 time steps, with the simulation time step set to $0.02\,\mathrm{s}$. Two Q-networks were used, each with two hidden layers of 256 units and ReLU activation. The policy was evaluated every few thousand steps during training to ensure stable learning. After training, the final policy was tested in a deterministic rollout of 500 simulation steps, and the resulting quaternion, angular velocity, and control torque histories were recorded for analysis.

Regarding the inertia parameters of the spacecraft, in practical ADR missions, the combined spacecraft's inertia is often uncertain due to the unknown mass distribution of the captured target. Here, the control problem is formulated as stabilizing a rigid spacecraft with unknown inertia. The model-free reinforcement learning algorithms do not require knowledge of the inertia matrix, learning the policy through interaction with the dynamics. For simulation, a fixed inertia matrix with $\mathbf{J} = \mathrm{diag}(3.0,\, 2.0,\, 1.0)\ \mathrm{kg \cdot m^2}$ is used to generate state transitions, but this information is not given to the learning algorithm.

The simulation used a fixed integration step size $\Delta t = 0.02\ \mathrm{s}$. The target attitude was the unit quaternion $\mathbf{q}_{\mathrm{target}} = [1,\, 0,\, 0,\, 0]^{\mathsf{T}}$, corresponding to zero rotation, and the target angular velocity was $\boldsymbol{\omega}_{\mathrm{target}} = \mathbf{0}\ \mathrm{rad/s}$. The initial state was set to a small rotation of $0.2\ \mathrm{rad}$ about each Euler axis, with an initial angular velocity $\boldsymbol{\omega}(0) = [0.1,\, 0.2,\, -0.1]^{\mathsf{T}}\ \mathrm{rad/s}$.

The control torque was bounded by $\|\boldsymbol{u}\|_\infty \leq 5.0\ \mathrm{N \cdot m}$. The reward function and its error terms were defined as

$$r = -(k_{\mathrm{att}}\, e_{\mathrm{att}} + k_{\mathrm{rate}}\, e_{\mathrm{rate}} + k_{\mathrm{torque}}\, e_{\mathrm{torque}}), \tag{3.11a}$$

$$e_{\mathrm{att}} = 1 - q_0^2, \tag{3.11b}$$

$$e_{\mathrm{rate}} = \left\| \boldsymbol{\omega} - \boldsymbol{\omega}_{\mathrm{target}} \right\|^2, \tag{3.11c}$$

$$e_{\mathrm{torque}} = \|\boldsymbol{u}\|^2. \tag{3.11d}$$

where r denotes the instantaneous reward, and e_{att}, e_{rate}, and e_{torque} represent the attitude, angular rate, and control effort errors, respectively. The weighting coefficients k_{att}, k_{rate}, and k_{torque} determine the relative importance of each error term in the total reward. Here, q_0 is the scalar part of the unit quaternion $\boldsymbol{q} = [q_0, q_1, q_2, q_3]^{\mathsf{T}}$ representing the spacecraft attitude, $\boldsymbol{\omega}$ is the body angular velocity vector, $\boldsymbol{\omega}_{\mathrm{target}}$ is the desired angular velocity, which is set to zero for attitude stabilization control, and $\boldsymbol{u}$ is the control torque vector applied to the spacecraft. This setup was chosen to encourage accurate attitude tracking with minimal rate deviation and control usage.

The weighting coefficients values are $k_{\mathrm{att}} = 10.0$, $k_{\mathrm{rate}} = 1.0$, and $k_{\mathrm{torque}} = 0.1$. The coefficients are selected to reflect the control priorities of the spacecraft stabilization task. The attitude error term is assigned the largest weight to ensure that the learning process primarily focuses on reducing orientation error. The angular rate term provides damping and encourages the system to

approach the equilibrium with small rotational velocities, preventing oscillatory behavior. The torque penalty term is introduced with a smaller weight to discourage excessive control effort and to keep the learned policy within actuator limits.

In practice, the weights were chosen so that the three error terms have comparable numerical magnitudes during nominal system responses, while maintaining the dominance of the attitude stabilization objective. Preliminary simulations were conducted to verify that the selected weights lead to stable learning and effective closed-loop regulation.

3.4. Critic match loss landscape visualization results

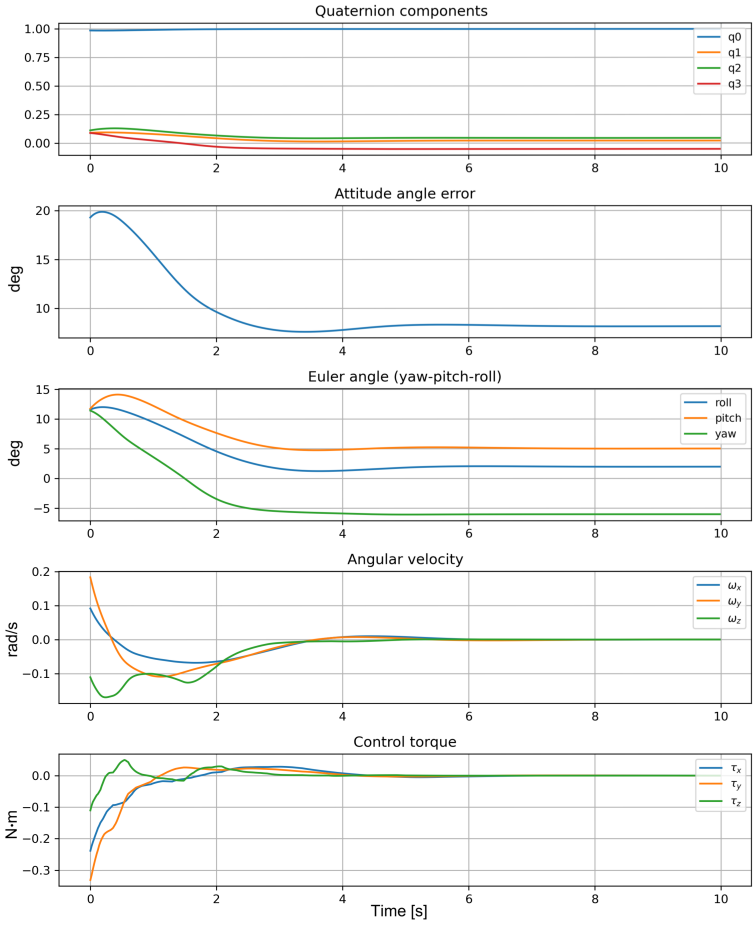
This section presents the critic match loss landscape analysis across multiple control scenarios to examine its interpretive capability and methodological scope. We first demonstrate the landscape corresponding to a convergent SAC control case, followed by a comparison between convergent SAC and divergent ADHDP to highlight structural differences under distinct learning paradigms. Next, a divergent SAC case is analyzed together with temporal landscape snapshots to investigate instability events during training. A direct comparison between convergent and divergent SAC cases is then conducted to examine how critic optimization trajectories differ under successful and unsuccessful control outcomes. Finally, we clarify the interpretation scope and methodological boundaries of the proposed visualization framework as a geometric diagnostic tool for critic optimization behavior. It needs to be mentioned that, in the experiment, both the visualization and the quantitative indices computation use a scaled loss to improve numerical readability. Since the scaling is linear, it does not alter the geometric characteristics of the loss landscape.

3.4.1. Final critic match loss landscape of convergent SAC control

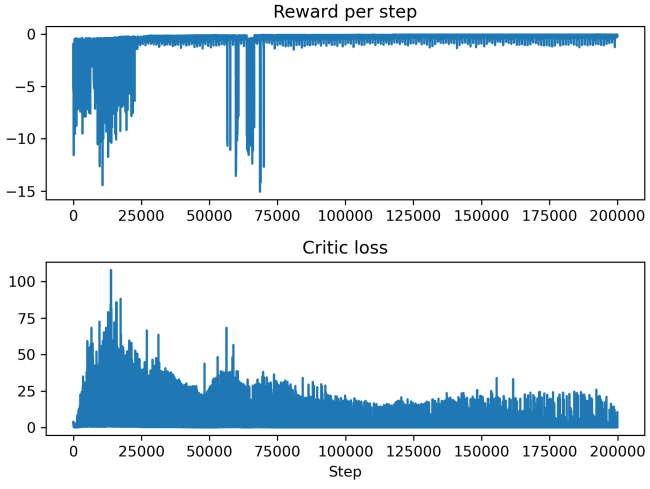
For the simulation of convergent SAC control, the main hyperparameters followed the default values in Stable-Baselines3: a learning rate of 3×10^{-4}, a replay buffer size of 10^6, a batch size of 256, $\gamma = 0.99$, $\tau_{\text{target}} = 0.005$, and automatic entropy tuning enabled. Actor and critic networks were updated at every step with 1 gradient iteration.

Figure 3.2 shows the spacecraft system performance and the training process of the SAC algorithm. From Figure 3.2a, it can be seen that the quaternions are stabilized around the given control target, which is $[1, 0, 0, 0]^{\mathrm{T}}$. Under the given control torque limit, the angular velocity is decreased from the initial state to around zero. These results confirm that the SAC policy successfully achieves stable spacecraft attitude control.

From Figure 3.2b, it can be seen that the reward is accumulated through training. The critic loss, which is initially large and fluctuating, gradually decreases and stabilizes, reflecting the convergence of Q-value estimation under



(a) SAC control result

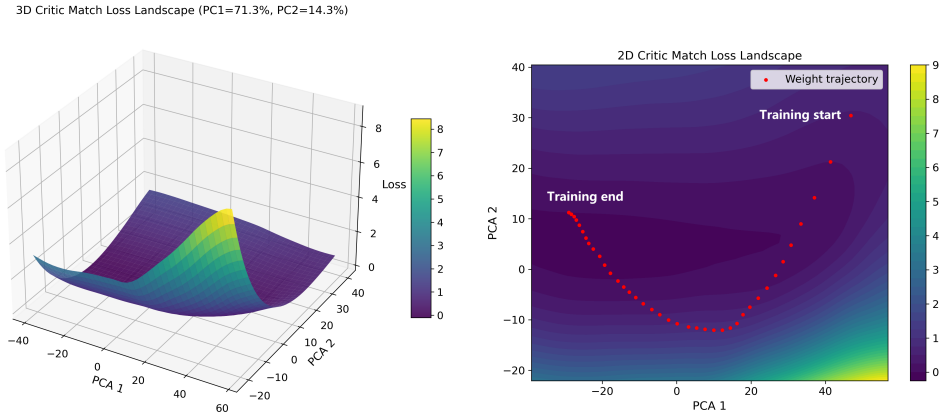


(b) SAC training curve

Figure 3.2: SAC convergent control results for spacecraft attitude system

the entropy-regularized objective of SAC. The training behavior demonstrates effective policy learning and critic approximation during training.

Using the method in subsection 3.2.3, the loss landscape under the final policy is generated for the SAC control process in section 3.3. To illustrate the result, only one critic network is selected to show the loss landscape during training. The weight is saved every 5000 steps during training. Data with batch size of 64 is generated as the final rollout after training is finished. Following this setting, the loss landscape of SAC algrotihm is shown in Figure 3.3.



(a) SAC 3-D loss landscape

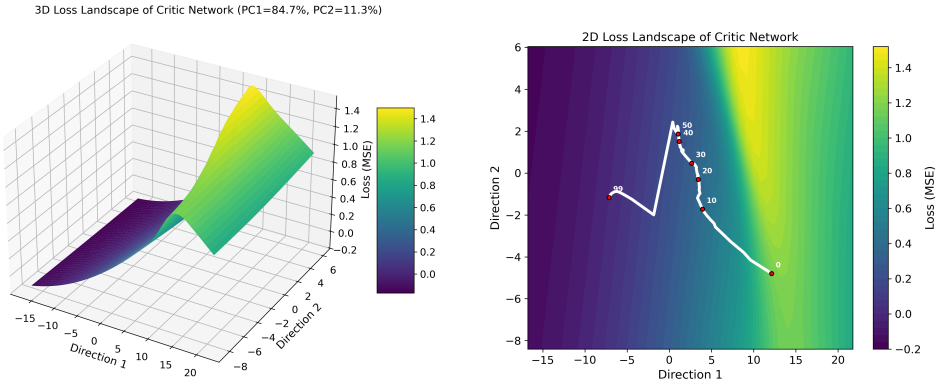
(b) SAC 2-D loss curve

Figure 3.3: Final 3-D and 2-D loss landscape of convergent spacecraft attitude SAC control

From Figure 3.3a, we can see that, the 3D loss landscape of SAC exhibits a broad, smooth basin with gentle curvature. It means small perturbations of the critic parameters produce limited variation in the match loss. This structure indicates robust convergence and insensitivity to local noise, which is benefited from the batch learning and entropy regularization technique that SAC employs. Figure 3.3b illustrates the optimization path of the critic parameters over training. The trajectory lies in a continuous valley. The earlier iterations span in a wider region, while the later path converges into a narrow and stable zone. This evolution is consistent with the critic loss curve in Figure 3.2b, which also shows steep loss reduction in the early stage and stabilization near the bottom of the basin.

3.4.2. Loss landscape comparison between convergent SAC control and divergent ADHDP control

Using the same critic match loss landscape construction framework, Figure 3.4 illustrates the 3-D loss landscape and the 2-D loss path resulting from applying the ADHDP algorithm to the spacecraft attitude system using with



(a) 3-D loss landscape

(b) 2-D loss landscape

Figure 3.4: 3-D and 2-D loss landscape of spacecraft attitude ADHDP control

quaternion expression. The control results corresponding to the ADHDP critic loss landscape are shown in Appendix C.

Table 3.1 shows the quantitative indices values of the landscapes of SAC and ADHDP algorithm on the spacecraft attitude control problem. ρ value for calculating sharpness is selected as 0.25. The ϵ value for calculating basin area is selected as 0.1.

Table 3.1: Quantitative loss landscape comparison between SAC and ADHDP

Metric	**SAC**	**ADHDP**
Sharpness (Sharp_ϵ)	0.725596	0.108401
Basin (A_ρ)	1252.259644	Non-existent
Local anisotropy ($\log \kappa$)	3.718889	10.390625

From the comparison between SAC's critic match loss landscape and ADHDP's landscape, there are three aspects that indicate obvious differences in the landscape of SAC algorithm and ADHDP algorithm.

The first aspect is the PCA range and the range of the optimization trajectory. For the ADHDP landscape, the small changes in the PCA direction indicate that the main direction of weight changes is relatively limited. It oscillates within a narrow parameter space. This often suggests limited exploration or oscillation within a restricted parameter manifold. In contrast, for the landscape of SAC algorithms, it has larger spans of weight updates in the main direction. The stronger contributions from the PCA principal components indicate that the network has explored a wider range of weight spaces and ultimately reached a low-loss region. Rather than indicating successful control solely through span magnitude, this difference highlights distinct optimization behaviors under online and replay-based training mechanisms.

The second aspect is the curvature and structure of the loss landscape. The ADHDP loss landscape exhibits sharp curvature and irregular contour shapes. This observation is consistent with the quantitative metrics in Table 3.1. The relatively high local anisotropy value $\log \kappa = 10.39$ indicates strong directional imbalance in curvature, meaning that gradients can change abruptly depending on direction. Moreover, the absence of a well-defined basin suggests that no stable low-loss region is formed. Together, these characteristics reflect unstable gradient behavior and sensitivity to perturbations during training. By contrast, the SAC loss landscape shows a clear concave structure with a single valley. Within this valley, the surface is relatively flat and wide. From Table 3.1, this valley has a large basin area with $A_\rho = 1252.26$ and a much lower anisotropy value with $\log \kappa = 3.72$, indicating more balanced curvature across directions. Although the sharpness value of SAC is higher than that of ADHDP, it is linked with a structured descent toward a well-defined basin rather than irregular local fluctuations. Such a geometry is linked with a more structured optimization region under the frozen-target objective, where parameter updates are less sensitive to directional imbalance compared to the ADHDP case.

The third aspect is the range of the loss. In the loss landscape, the scale of variation between the highest and lowest loss for the ADHDP algorithm is considerably smaller than that for the SAC algorithm. It means that, during training of the ADHDP algorithm, it explores within a limited zone, of which the loss is not so much different. However, for the SAC algorithm, the large changes in loss indicate that the optimization goes through a more complete procedure, which reflects a more extended scan of the solution subspace. It has to be noted that, the presence of negative values in the normalized loss surface of ADHDP indicates that the final critic parameters are not located at a local minimum within the scanned subspace. Consequently, the loss landscape does not exhibit a closed basin structure around the final point.

These results show that the proposed critic match loss landscape visualization method remains applicable in the off-policy reinforcement learning setting. The smooth basin revealed in the SAC critic match loss landscape corresponds to a smooth optimization region and a convergent critic approximation, which is consistent with SAC's replay-based updates and target smoothing technique. In contrast, the sharper geometry observed in the ADHDP landscape shows the sensitivity of online adaptive learning to data noise and unstable target updates. The comparison between the two algorithms shows that the proposed visualization method can interpret the critic learning stability for actor–critic reinforcement learning algorithms. It also demonstrates that the method remains effective across different training paradigms.

3.4.3. Temporal evolution of critic optimization of divergent SAC control

The geometric characteristics of critic optimization of the convergent SAC case under the proposed visualization framework is illustrated in subsection 3.4.1.

To further examine how the critic match loss landscape behaves under different training outcomes within the same algorithmic structure, we next consider two divergent SAC cases obtained under altered training conditions. To investigate how the geometry develops over time, critic match loss landscapes are constructed at multiple training stages. The temporal evolution of the projected critic parameter trajectory is then examined to reveal changes in optimization behavior as divergence emerges.

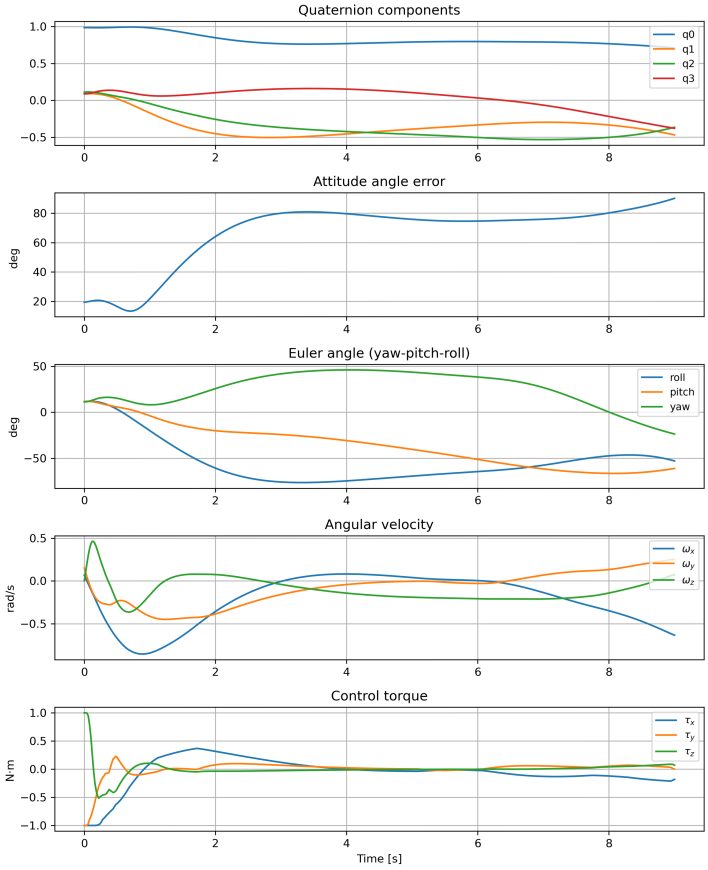
For the divergent-control experiments, the learning rate was increased to 2×10^{-1}, the replay buffer size was reduced to 5×10^{2}, and the batch size was set to 16. A large discount factor $\gamma = 0.999$ and a fast target update coefficient $\tau_{\text{target}} = 0.999$ were used. Entropy regularization was fixed to 10^{-3} without automatic tuning. Actor and critic networks were updated at every step with 10 gradient iterations. This configuration produced divergent attitude responses.

Divergent SAC with critic parameter divergence

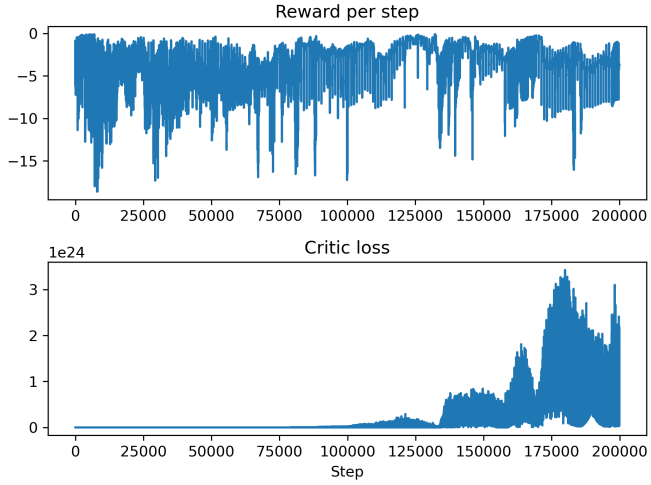
We now examine a divergent SAC realization in which critic instability develops with pronounced parameter divergence. While the algorithmic structure remains unchanged, this training outcome exhibits fundamentally different optimization behavior within the critic network.

Figure 3.5 shows the final rollout trajectory of the attitude system and the training curves. From the closed-loop perspective, the final rollout demonstrates clear failure of attitude stabilization. The attitude angle error increases over time rather than converging toward zero, and the angular velocity components do not decay. The control torque signals do not form a stabilizing feedback pattern, indicating that the learned policy fails to stablize the spacecraft dynamics. The scalar training curves further reveal instability. The critic loss grows to extremely large magnitudes during later stages of training, and the reward does not exhibit sustained improvement. Figure 3.6 illustrates the critic weight and actor weight update with training steps. Consistently, the critic weight norm increases almost monotonically throughout training and reaches a significantly larger scale than in the convergent case. In contrast, the actor weight norm grows more moderately and eventually stabilizes. This divergence of critic parameter magnitude suggests directional runaway rather than bounded oscillation. However, scalar curves alone do not reveal along which parameter directions this instability develops, nor how it relates to the geometry of the optimization landscape.

The final 3D critic match loss landscape in Figure 3.7a provides essential geometric insight. Under the frozen-target approximation, the surface does not exhibit a well-defined closed basin structure. Instead, it resembles a strongly anisotropic slope or ridge extending along the dominant principal direction. The PCA variance analysis indicates that the first principal component accounts for 96.5% of the total variance, while the second component contributes only marginally. This extreme dominance of PC1 implies that critic parameter evolution is effectively confined to a nearly one-dimensional subspace. In such a geometrically degenerate setting, parameter updates are driven primarily

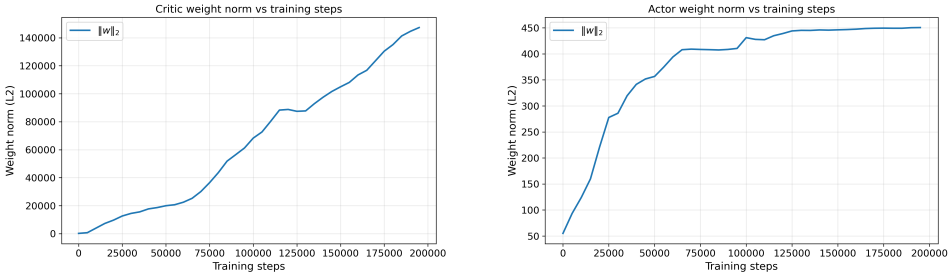


(a) SAC control result



(b) SAC training curve

Figure 3.5: Divergent control results for spacecraft attitude system under SAC with divergent critic



(a) critic weight with training steps

(b) actor weight with training steps

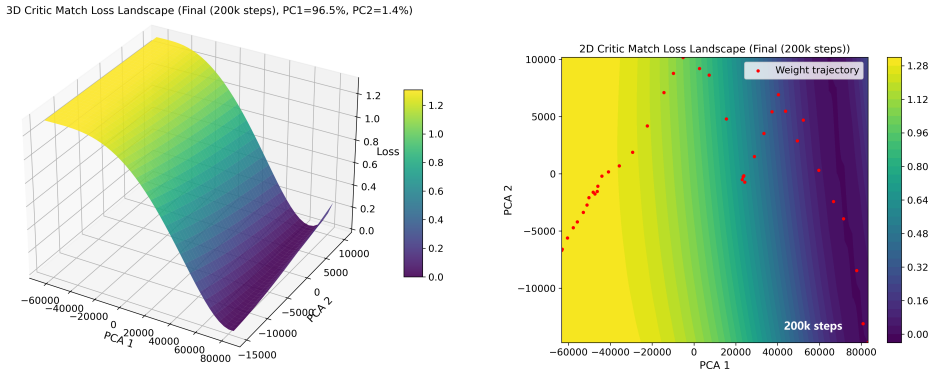
Figure 3.6: Critic and actor weight with training steps for SAC with divergent critic

along a single direction. If this dominant direction does not correspond to a stable descent toward a basin, persistent amplification along it can lead to uncontrolled parameter growth. Importantly, this near one-dimensional update behavior is not directly observable from the scalar weight-norm curve. It becomes evident only through the PCA structure of the loss landscape.

The corresponding 2D projection in Figure 3.7b further clarifies the parameter evolution. The loss contours lack closed low-loss regions indicative of a stable basin. The projected critic trajectory spans a large distance primarily along the PC1 direction, with limited variation along PC2. Rather than exhibiting discrete regime transitions between separated clusters, the trajectory displays sustained directional drift. This behavior is consistent with the PCA variance structure and explains the monotonic increase in critic weight norm that the optimization process is not exploring multiple regions of the landscape, but instead being progressively driven along a dominant and geometrically sensitive axis.

To interpret the temporal snapshots in Figure 3.8, it is important to recall that the PCA plane is constructed using critic weights collected over the entire training process. All snapshots are therefore projected onto a common global coordinate frame, ensuring comparability of trajectory evolution. The reported variance ratio remains constant across snapshots.Because this global PCA basis is influenced by late-stage parameter excursions, early-stage grid points may include regions far from the contemporaneous parameter neighborhood. Consequently, the temporal interpretation should not rely on absolute colorbar ranges. Instead, emphasis is placed on trajectory alignment and geometric structure. Figure 3.8 demonstrates temporal snapshots of the critic match loss landscape which are generated at 20k, 50k, 100k, and 200k steps. Across snapshots, the critic trajectory continues to move along the same dominant principal direction, without forming a closed basin, suggesting sustained directional growth rather than stable convergence.

Overall, the final and temporal critic match loss landscapes of the divergent SAC control with divergent critic parameters demonstrate that when the critic



(a) 3-D loss landscape for divergent SAC control with divergent critic

(b) 2-D loss curve for divergent SAC control with divergent critic

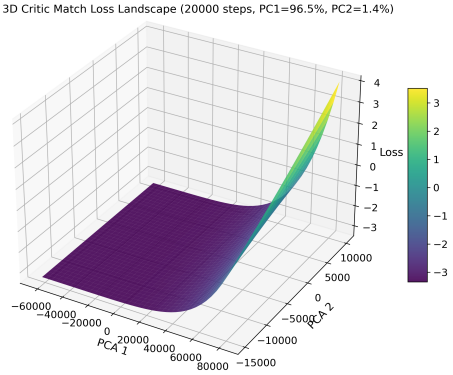
Figure 3.7: 3-D and 2-D loss landscape of divergent spacecraft control under SAC with divergent critic

match loss landscape exhibits strong anisotropy and effective dimensional collapse in parameter space, critic optimization may become unstable and lead to runaway parameter growth. In this case, the geometrically degenerate structure of the frozen-target landscape aligns with divergent critic parameters and closed-loop control failure. The critic match loss visualization therefore serves as a geometric diagnostic tool, revealing structural instability in the critic optimization process that is not directly visible from scalar training curves alone.

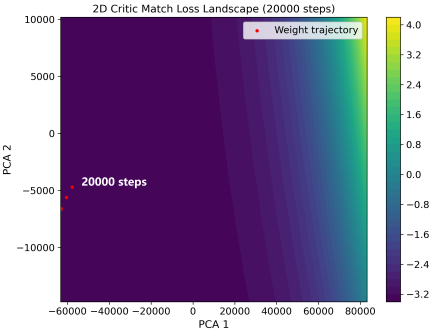
Divergent SAC with critic parameter convergence

In the divergent SAC case, as indicated by Figure 3.9a, the spacecraft attitude does not converge to the desired equilibrium. The final rollout shows that the attitude angle error increases over time rather than decreasing. The angular velocity components do not decay toward zero, and a clear drift is observed along the $\bar{z}$ axis, or the $\overline{n_3}$ axis as mentioned in Appendix A. The corresponding control torque remains close to its boundary for periods, indicating that the learned policy does not establish a stabilizing feedback law. Closed-loop control therefore fails.

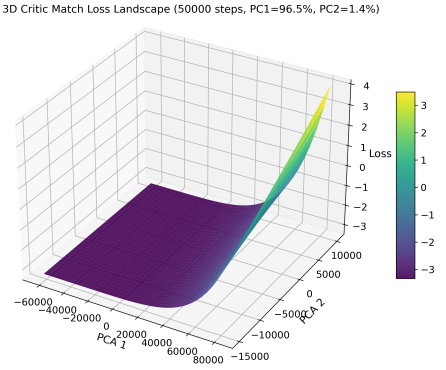
The scalar training curves in Figure 3.9b provide additional evidence of instability. Although the critic loss eventually decreases to a relatively small value, several pronounced spikes occur during training. One large spike appears around 30k steps, and another significant excursion is observed near 100k steps. These spikes are substantially higher than surrounding values and reflect abrupt transitions in critic update dynamics rather than gradual convergence. The reward curve also remains persistently unfavorable, without a clear monotonic improvement trend.



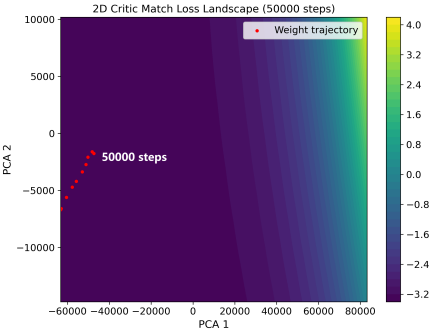
(a) 3-D landscape at step 20000



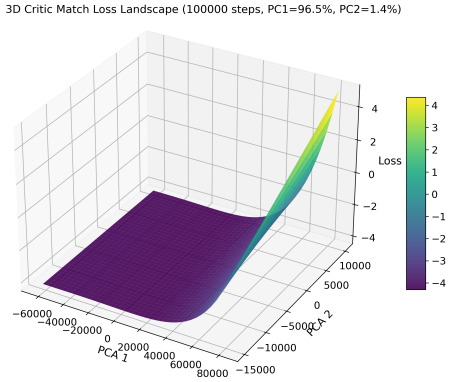
(b) 2-D landscape at step 20000



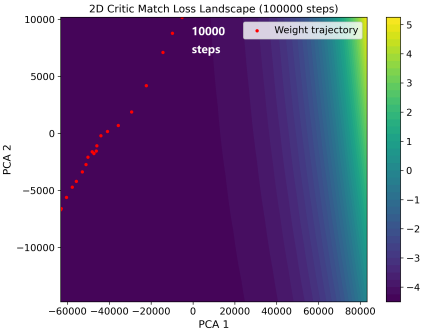
(c) 3-D landscape at step 50000



(d) 2-D landscape at step 50000

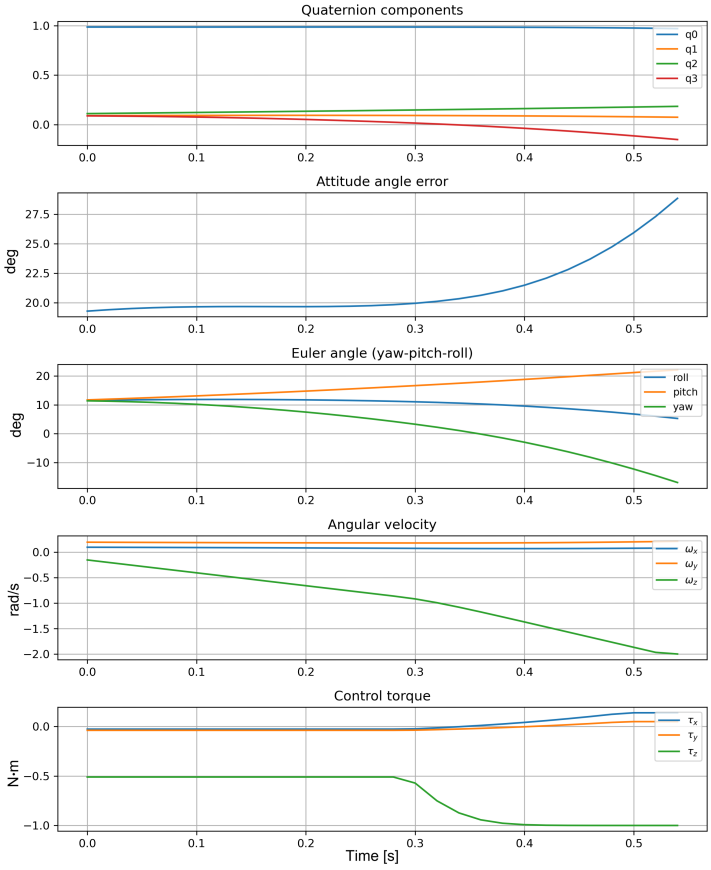


(e) 3-D landscape at step 100000

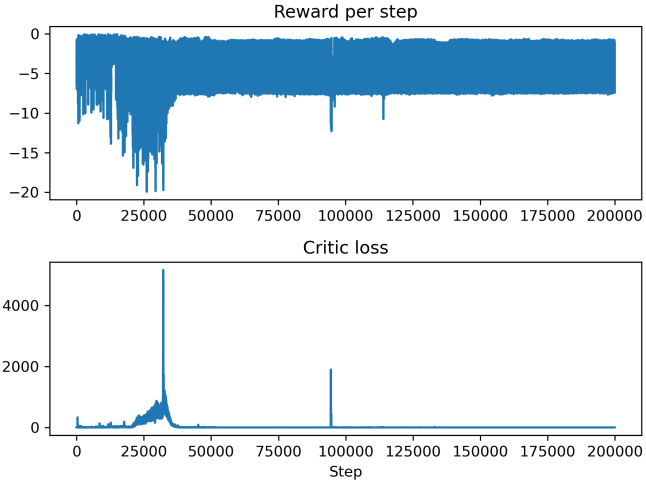


(f) 2-D landscape at step 100000

Figure 3.8: Snapshots during training of 3-D and 2-D loss landscape of divergent spacecraft attitude control under SAC with divergent critic



(a) SAC control result



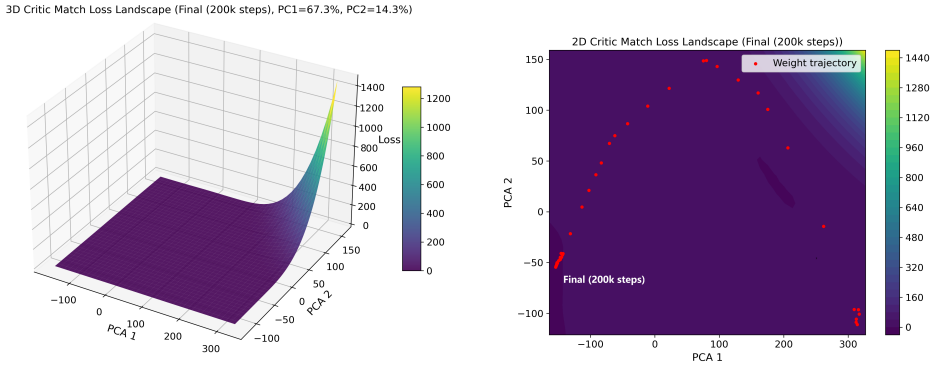
(b) SAC training curve

Figure 3.9: Divergent control results for spacecraft attitude system under SAC with convergent critic

To interpret these behaviors from the perspective of critic optimization, the critic match loss landscape is evaluated at the final stage, which corresponds to 200k steps, under a fixed replay batch and frozen target. As indicated by Figure 3.10, the final landscape exhibits broad smooth regions in the PCA plane together with steep high-loss walls in certain directions. The projected critic parameter trajectory does not form a single continuous curve. Instead, it appears separated into distinct clusters with large spatial gaps between them. The presence of separated clusters suggests that the critic underwent multiple discontinuous transitions during training rather than evolving smoothly toward a single stable region.

Importantly, the final landscape topology itself does not appear fundamentally different from convergent SAC cases. The failure therefore cannot be attributed to a rugged critic geometry. Instead, instability arises from discontinuities in the optimization path. The critic evolves across separated regions of the critic match loss landscape, leading to regime shifts that are not apparent from scalar loss values alone.

To further examine the optimization dynamics, Figure 3.11 demonstrates temporal snapshots of the critic match loss landscape which are generated at 20k, 50k, 100k, and 200k steps. The temporal sequence of critic match loss landscapes provides insight into the evolution of critic optimization geometry under the frozen-target approximation.

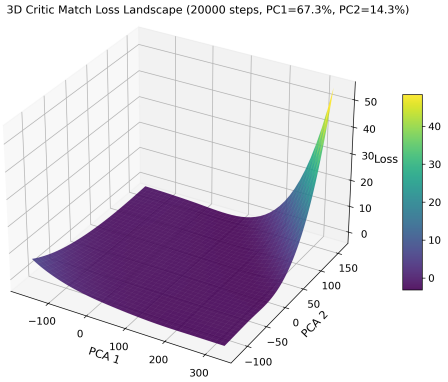


(a) 3-D loss landscape for divergent SAC control

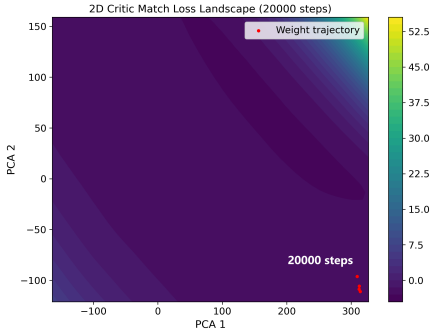
(b) 2-D loss curve for divergent SAC control

Figure 3.10: 3-D and 2-D loss landscape of divergent spacecraft attitude control under SAC with convergent critic

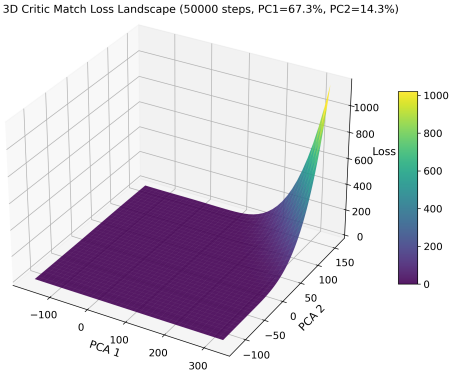
Figure 3.11 further clarify this mechanism. At 20k steps, the trajectory remains concentrated within a compact region of the PCA plane, indicating coherent parameter refinement. Around 50k steps, following the first major critic loss spike in the scalar critic loss curve, the projected trajectory expands and separates into distant regions, suggesting a substantial parameter



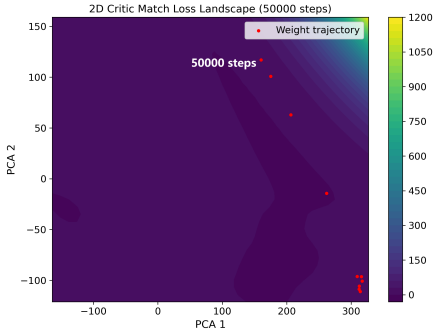
(a) 3-D landscape at step 20000



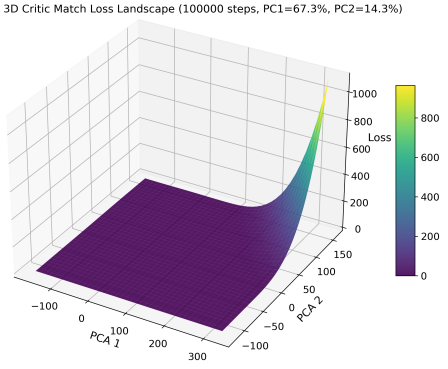
(b) 2-D landscape at step 20000



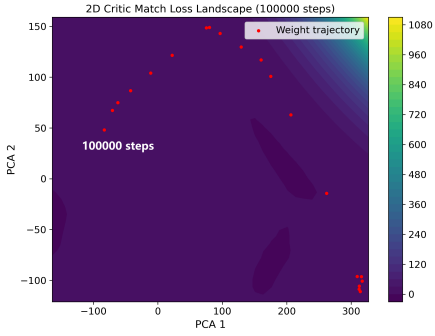
(c) 3-D landscape at step 50000



(d) 2-D landscape at step 50000



(e) 3-D landscape at step 100000



(f) 2-D landscape at step 100000

Figure 3.11: Snapshots during training of 3-D and 2-D loss landscape of divergent spacecraft attitude control under SAC with convergent critic

relocation. At 100k and 200k steps, although the overall 3D surface topology remains qualitatively similar, the trajectory remains fragmented across distinct clusters. The geometry is largely preserved, but the optimization path becomes discontinuous.By 200k steps, the clustering pattern persists. The final critic parameters locate within one geometrically structured region that appears locally stable under the frozen objective, yet remains disconnected from earlier regions explored during training. The trajectory history therefore does not represent a continuous refinement toward a single low-loss basin, but rather a sequence of regime shifts across the frozen-target geometry.

An important observation is that the global 3D surface topology does not change dramatically across temporal snapshots, Under the frozen-target approximation, even when the closed-loop behavior ultimately fails, the critic often evolves within geometrically structured and locally stable regions. This suggests that the divergent SAC case may correspond to a geometrically stable yet ineffective control solution at multiple stages of training.

These observations indicate that, in the case of divergent SAC control with convergent critic parameters, closed-loop failure does not stem from an ill loss surface, but from unstable critic optimization dynamics developed as trajectory discontinuities. The critic match loss landscape therefore provides geometric context for understanding optimization stability, rather than serving as a direct predictor of control convergence.

3.4.4. Comparison between convergent and divergent SAC control

A direct comparison among the convergent SAC case and the two divergent SAC cases reveals several important distinctions in critic optimization geometry.

First, the static 3D landscape alone cannot serve as a definitive indicator of closed-loop success. In both the convergent SAC case and the divergent SAC case with a smooth critic geometry, the final frozen-target landscape exhibits broad smooth regions and coherent basin-like structures. The global topology does not appear pathological in either case. This confirms that a geometrically structured critic surface does not guarantee successful control.

In the divergent SAC case with divergent critic parameters, the key geometric feature is strong directional domination. The PCA analysis shows that nearly all variance is captured by the first principal component, indicating that critic updates are effectively confined to a single dominant direction. The 3-D surface resembles an elongated ridge rather than a balanced basin, and the projected trajectory extends predominantly along this direction. This directional phenomenon is not evident from the scalar weight-norm curve alone. Only through PCA variance ratios and 2-D projected trajectories does the underlying instability mechanism become visible.

In contrast, the divergent SAC case with convergent critic geometry exhibits a smooth final surface without pronounced anisotropy. However, the projected trajectory reveals discontinuous parameter relocations across separated regions of the PCA plane. Here, instability is not caused by geometric degeneration of the surface itself, but by abrupt transitions in optimization dynamics. These

transitions may be associated with interactions between critic updates and other components of the actor–critic architecture, including policy saturation effects, exploration dynamics, reward scaling, or replay-induced distribution shifts, which require further investigation.

This distinction becomes clearer when examining weight dispersion in the PCA plane. In the convergent SAC case, dispersion is moderate and progressively concentrates within a coherent low-loss region, reflecting stable refinement. In the divergent SAC case with convergent critic geometry, dispersion appears as fragmented clusters, consistent with discontinuous parameter relocations across separated regions. In the divergent SAC case with divergent critic parameters, dispersion is not clustered but directionally amplified along a dominant principal component, revealing anisotropic instability. In contrast, the divergent ADHDP case exhibits narrow dispersion, suggesting limited weight exploration rather than convergence. These comparisons indicate that dispersion magnitude alone is insufficient, while its geometric structure and relation to trajectory continuity determine its interpretive value.

These observations clarify the interpretation of critic match loss visualization. The method should not be used as a binary indicator of control convergence based solely on surface shape. Instead, it provides a geometric diagnostic framework.By integrating the 3-D loss geometry and the projected 2-D optimization trajectory, the method characterizes the evolution of critic parameters within the loss landscape. It enables the identification of distinct optimization patterns, including dominance by a single unstable direction or fragmentation across disconnected regions. This combined geometric and trajectory analysis clarifies how the critic evolves during training and explains the practical value of the critic match loss landscape.

3.4.5. Interpretation scope and methodological boundaries of the frozen-target landscape

The critic match loss landscape presented in this work is constructed under a frozen-target approximation. By fixing the replay batch and precomputing target values, the dynamic Bellman update process is locally transformed into a stationary surrogate objective. While this simplification is necessary to obtain a consistent and plottable surface, it introduces an inherent gap between the static landscape and the fully non-stationary optimization process encountered during reinforcement learning.

First, the critic match loss evaluated on the frozen-target surface does not coincide exactly with the instantaneous critic loss recorded during training. The true critic loss evolves with changing targets, replay distributions, and policy updates, whereas the match loss is computed under fixed targets associated with a specific policy snapshot. As a result, the weight trajectory projected onto the PCA plane does not correspond one-to-one with the scalar critic loss curve. Nevertheless, the frozen-target landscape remains informative in revealing structural properties of parameter evolution, such as the presence

of dominant weight update, clustering behavior, and transitions across regions of varying geometric sensitivity. Even though the absolute loss values differ, these geometric events often align temporally with significant fluctuations in the scalar training curves,

Second, the static landscape does not represent the global loss surface over all possible policies or target configurations. It reflects only the geometry under a specific frozen policy and replay batch. Consequently, it cannot evaluate the critic performance across the full dynamic distribution shift induced by actor–critic coupling. To partially address this limitation, temporal snapshots are introduced. By constructing frozen-target landscapes at multiple training stages, the method captures how the effective local geometry evolves over time. Although each snapshot remains static, the sequence of landscapes provides insight into regime transitions and structural stability during training.

The interpretability strength of this frozen-target geometry depends on the structural complexity of the underlying algorithm. In online methods such as ADHDP, critic updates are tightly coupled with policy behavior and target computation, and relatively few auxiliary mechanisms need to be frozen. Consequently, the frozen-target objective more closely resembles the effective training objective, and geometric instability of the critic is directly reflected in control performance. In contrast, SAC incorporates replay buffering, target networks, twin critics, and entropy regularization, all of which partially decouple critic optimization from immediate closed-loop behavior. As a result, a larger set of components must be frozen to define a stationary surrogate objective, and the geometric stability of the critic under this surrogate does not necessarily imply control convergence. This difference arises primarily from architectural complexity rather than from the freezing procedure itself.

In conclusion, these considerations clarify the interpretation scope of the critic match loss visualization. The static landscape should not be viewed as a direct reconstruction of the full dynamic reinforcement learning objective. Although closed-loop performance is governed by broader interactions within the actor–critic system, the critic match loss landscape serves as a geometric diagnostic tool that characterizes critic optimization behavior at selected training stages. Its value lies in revealing structural properties of the loss geometry, directional dominance in parameter updates, and discontinuous transitions in weight.

3.5. Validation on benchmark system

To validate the proposed critic match loss landscape visualization method for off-policy reinforcement learning algorithms, the continuous cart-pole system is selected as a benchmark to further demonstrate the interpretation procedure of the method.

The dynamics of the continuous cart-pole system is given in detail in B. The system was trained using the Soft Actor–Critic (SAC) algorithm from the Stable-Baselines3 library [15]. The actor and critic networks adopt the

default multi-layer perceptron (MLP) architecture with twin Q-functions and a stochastic Gaussian policy. Automatic entropy tuning is enabled to adaptively adjust the temperature parameter. The learning rate is set to 3×10^{-4} for all networks. Training is performed for 5×10^4 interaction steps with simulation step size $0.02\,\mathrm{s}$. A replay buffer of size 10^5 stores past transitions, from which mini-batches of 256 samples are drawn. Learning begins after the first 10^3 collected samples. The discount factor and soft target update coefficient are chosen as 0.99 and 0.005, respectively, and one gradient update is executed per environment step.

To encourage stable upright balancing and smooth control, a quadratic cost structure is adopted. Let the system state be $s = [\psi, \dot{\psi}, h, \dot{h}]^\top$ and the normalized control action be a. The instantaneous cost is defined as

$$c(s,a) = w_\psi \psi^2 + w_{\dot{\psi}} \dot{\psi}^2 + w_h \left(\frac{h}{h_{\max}}\right)^2 + w_{\dot{h}} \dot{h}^2 + w_u a^2, \tag{3.12}$$

where ψ and $\dot{\psi}$ are converted to radians, $h_{\max}$ denotes the track limit, and $w_\psi, w_{\dot{\psi}}, w_h, w_{\dot{h}}, w_u$ are positive weighting coefficients.

In this work, the weights are empirically selected as

$$w_\psi = 5.0, \quad w_{\dot{\psi}} = 0.1, \quad w_h = 1.0, \quad w_{\dot{h}} = 0.1, \quad w_u = 0.01. \tag{3.13}$$

The reward is defined as the negative cost,

$$r(s,a) = -c(s,a), \tag{3.14}$$

which encourages the pole to remain upright, suppresses excessive velocities, keeps the cart near the center of the track, and penalizes aggressive control inputs.

Additionally, a large terminal penalty is imposed when safety limits are violated,

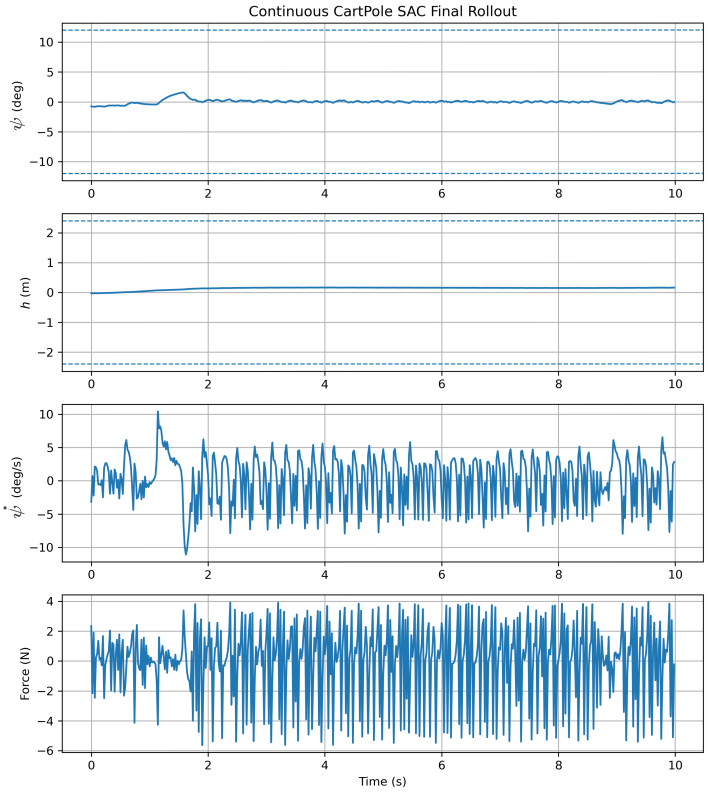
$$r_{\text{fail}} = -50, \tag{3.15}$$

if $|h| > h_{\max}$ or $|\psi| > \psi_{\max}$. This extra penalty provides a strong learning signal to discourage failure states.

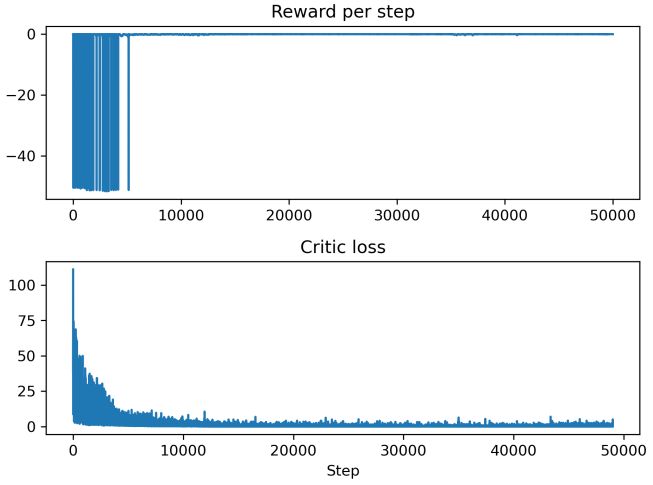
After training, the learned policy is evaluated using a deterministic rollout of 500 simulation steps. The resulting pole angle, angular velocity, cart displacement, applied control forces and the training curves are shown in Figure 3.12.

The training curves of the SAC algorithm on the continuous cart-pole system are shown in Figure 3.12b. The reward rapidly increases from large negative values and stabilizes near zero, indicating successful pole stabilization and cart position regulation. Meanwhile, the critic loss decreases monotonically during the early training phase and gradually converges to a small and stable range without significant oscillations. This behavior suggests stable critic approximation and consistent Bellman error minimization.

The final rollout of the cart-pole system further confirms successful control performance. The pole angle remains close to the upright equilibrium, the cart



(a) SAC control cart-pole result



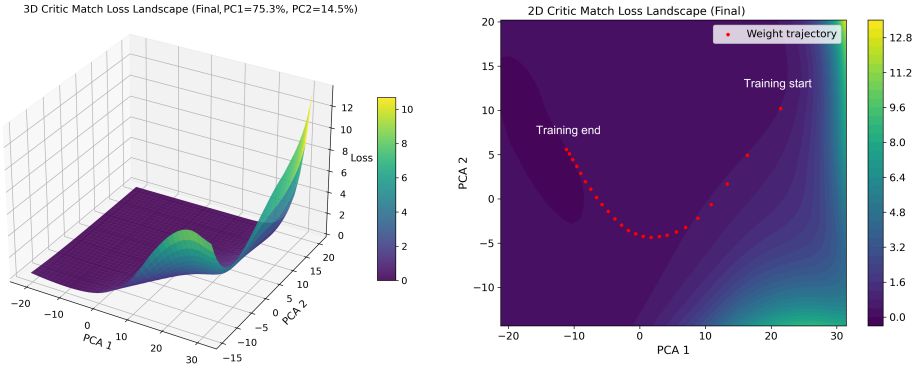
(b) SAC cart-pole training curve

Figure 3.12: SAC control results for cart-pole system

position stays within bounded limits, and the control force remains finite and well-regulated. These results demonstrate that the SAC algorithm achieves stable convergence in this benchmark environment

The corresponding critic match loss landscape is presented in Figure 3.13. Similar to the convergent spacecraft case, the loss surface exhibits a clear concave valley structure with a well-defined low-loss basin. When projected onto the PCA plane, the recorded critic weight trajectory exhibits a gradual progression toward the low-loss region defined by the frozen-target objective, with the final parameters situated inside the basin. The surface curvature is smooth and does not display irregular spikes or fragmented contour patterns.

Although the cart-pole system differs significantly from spacecraft attitude dynamics in state dimension and physical structure, the geometric characteristics of the critic match loss landscape remain consistent with those observed in the convergent spacecraft case. This consistency suggests that the relationship between landscape geometry and critic convergence is not specific to a particular dynamical system, but reflects a more general property of stable off-policy actor–critic training.



(a) SAC cp 3-D loss landscape

(b) SAC cp 2-D loss curve

Figure 3.13: 3-D and 2-D loss landscape of cart-pole SAC control

3.6. Conclusion

Off-policy reinforcement learning has a different structure from online reinforcement learning in data flow and target calculation in TD error. This work adapts the critic match loss landscape visualization method originally developed for online reinforcement learning to the off-policy reinforcement learning framework by explicitly addressing differences in data flow and target construction. The Soft Actor–Critic (SAC) algorithm is adopted as the demonstration case of the adapted critic match loss landscape method for off-policy RL. By applying SAC to the spacecraft attitude control problem, the critic

match loss landscape is reconstructed and compared with that of the ADHDP algorithm using sharpness, basin area, and local anisotropy metrics, together with temporal landscape snapshots. The results show that the convergent SAC case exhibits a coherent basin structure under the frozen-target objective, while the ADHDP case presents sharper and more irregular geometry. The divergent SAC case further demonstrates that geometric stability of the critic under the surrogate objective does not necessarily guarantee closed-loop convergence, reflecting the influence of broader algorithmic interactions. Overall, the proposed visualization framework serves as a geometric diagnostic tool for interpreting critic optimization behavior. Although it does not reconstruct the full non-stationary training objective, it provides a consistent approach for analyzing structural properties of critic learning across both online and off-policy actor–critic algorithms.

Appendix A: Attitude dynamics of spacecraft with unknown inertia

in this chapter, a body-fixed reference frame $\mathcal{B}$ is defined at the center of mass of the combined spacecraft, with its axes aligned to the principal directions of inertia. An Earth-Centered Inertial (ECI) frame $\mathcal{N}$ is also defined, with unit vectors $\{\overline{\boldsymbol{n_1}}, \overline{\boldsymbol{n_2}}, \overline{\boldsymbol{n_3}}\}$. The $\overline{\boldsymbol{n_1}}$ axis points toward the mean equinox, the $\overline{\boldsymbol{n_3}}$ axis is parallel to the Earth's rotation axis (celestial north), and $\overline{\boldsymbol{n_2}}$ completes the right-handed system.

The attitude of the combined spacecraft is represented by a unit quaternion

$$\boldsymbol{q} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{A.1}$$

which defines the rotation from $\mathcal{N}$ to $\mathcal{B}$. This formulation avoids singularities and is suitable for cases where the attitude may vary over a wide range. The angular velocity of the spacecraft expressed in frame $\mathcal{B}$ is denoted as

$$\boldsymbol{\omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}. \tag{A.2}$$

The kinematic equation is

$$\dot{\boldsymbol{q}} = \frac{1}{2}\, \boldsymbol{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix} \tag{A.3}$$

where $\otimes$ denotes the quaternion multiplication.

The attitude dynamics are expressed as

$$\boldsymbol{M} = \hat{J}_{\text{sc}} \cdot \dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \left(\hat{J}_{\text{sc}} \cdot \boldsymbol{\omega} \right) \tag{A.4}$$

Here, $\boldsymbol{M}$ is the control torque, and $\hat{J}_{\text{sc}}$ is the inertia matrix of the combined spacecraft, which is assumed to be unknown.

Appendix B: Dynamics of the continuous cart-pole cystem

A cart of mass m_c moves horizontally along a track, and an inverted pendulum of mass m is attached to the cart by a frictional pivot. The pendulum is constrained to rotate only in the plane. An idealized model is assumed in which the pendulum mass is concentrated at a single point located at the end of a rigid massless rod of length l.

The upright configuration corresponds to an unstable equilibrium. Without active control, gravity causes the pendulum to fall away from the vertical position. Therefore, an external horizontal force must be continuously applied to the cart in order to stabilize the pendulum. In contrast to the classical discrete cart-pole benchmark, where the control input takes values from a finite set, the present formulation adopts a *continuous control action*. Specifically, the controller can generate any real-valued force within a bounded interval, leading to smoother and more realistic actuation.

The system state is defined as

$$s = [\psi,\ \dot{\psi},\ h,\ \dot{h}]^\top, \tag{B.5}$$

where ψ denotes the angular displacement of the pole from the upright vertical direction, $\dot{\psi}$ is the angular velocity, h is the horizontal displacement of the cart, and $\dot{h}$ is the cart velocity.

The control input is a continuous scalar action

$$a \in [-1, 1], \tag{B.6}$$

which is linearly mapped to the physical force applied to the cart,

$$f_{\text{cp}} = aF_{\max}, \tag{B.7}$$

where $F_{\max}$ is the maximum allowable force magnitude.

Based on Newtonian mechanics, the nonlinear dynamics of the cart–pole system are described by

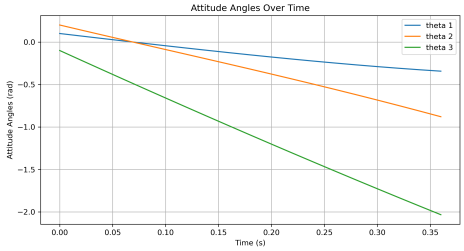
$$\ddot{\psi} = \frac{g\sin\psi + \cos\psi\left(\frac{-f_{\text{cp}} - ml\dot{\psi}^2\sin\psi}{m_c+m}\right)}{l\left(\frac{4}{3} - \frac{m\cos^2\psi}{m_c+m}\right)}, \tag{B.8}$$

$$\ddot{h} = \frac{f_{\text{cp}} + ml\left(\dot{\psi}^2\sin\psi - \ddot{\psi}\cos\psi\right)}{m_c + m}, \tag{B.9}$$

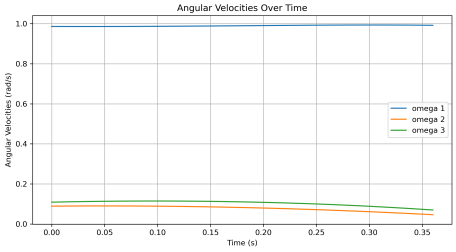
where $g = -9.8\,\text{m/s}^2$ denotes gravitational acceleration. Optional viscous or Coulomb friction terms for both the cart and the pendulum can be incorporated to account for mechanical dissipation.

The control objective is to learn a policy that continuously generates forces to maintain $\psi \approx 0$ (upright balance), suppress angular and translational velocities, keep the cart position within track limits and avoid excessive control effort. This continuous-action formulation enables smoother control signals and provides a more faithful representation of practical control systems compared with discrete force switching.

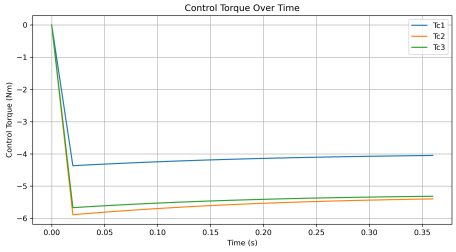
Appendix C: Control results of ADHDP for the spacecraft attitude system



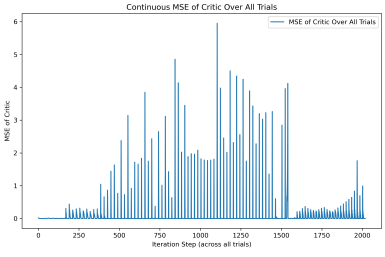
(a) Quaternion



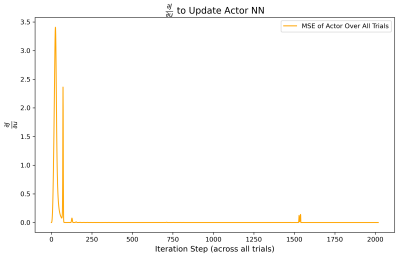
(b) Angular velocity



(c) Control torque



(d) ADHDP critic training curve



(e) ADHDP actor training curve

Figure 3.14: ADHDP control results for spacecraft attitude system

References

[1] H. Hu, W. Wu, J. Zhang, J. Wang, and Y. Song. “Robust trajectory maneuver scheduling near the flyby of small celestial bodies on the basis of proximal policy optimization”. In: *Astrodynamics* 9.6 (2025), pp. 855–875.

[2] B. Recht. “A tour of reinforcement learning: The view from continuous control”. In: *Annual Review of Control, Robotics, and Autonomous Systems* 2.1 (2019), pp. 253–279.

[3] W. Zhang, Y. Wang, and Y. Zhang. “Attention-driven reinforcement learning for multi-satellite collaborative orbital interception strategy solution”. In: *Astrodynamics* 9.5 (2025), pp. 657–669.

[4] A. Alharin, T.-N. Doan, and M. Sartipi. “Reinforcement Learning Interpretation Methods: A Survey”. In: *IEEE Access* 8 (2020), pp. 171058–171077. doi: 10.1109/ACCESS.2020.3023394.

[5] O. Bastani, Y. Pu, and A. Solar-Lezama. “Verifiable reinforcement learning via policy extraction”. In: *Advances in neural information processing systems* 31 (2018).

[6] B. Hayes and J. A. Shah. “Improving robot controller transparency through autonomous policy explanation”. In: *Proceedings of the 2017 ACM/IEEE international conference on human-robot interaction*. 2017, pp. 303–312.

[7] Q.-s. Zhang and S.-C. Zhu. “Visual interpretability for deep learning: a survey”. In: *Frontiers of Information Technology & Electronic Engineering* 19.1 (2018), pp. 27–39.

[8] I. Kostrikov, R. Fergus, J. Tompson, and O. Nachum. “Offline reinforcement learning with fisher divergence critic regularization”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 5774–5783.

[9] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein. “Visualizing the loss landscape of neural nets”. In: *Advances in neural information processing systems* 31 (2018).

[10] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor”. In: *International conference on machine learning*. Pmlr. 2018, pp. 1861–1870.

[11] Y. Sola, G. Le Chenadec, and B. Clement. “Simultaneous control and guidance of an auv based on soft actor–critic”. In: *Sensors* 22.16 (2022), p. 6072.

[12] M. Meijkamp. “Robust Attitude Control in Active Debris Removal Missions using Reinforcement Learning”. In: ().

[13] M. Shan, J. Guo, and E. Gill. “Review and comparison of active space debris capturing and removal methods”. In: *Progress in Aerospace Sciences* 80 (2016), pp. 18–32. issn: 0376-0421. doi: https://doi.org/10.1016/j.paerosci.2015.11.001. url: http://www.sciencedirect.com/science/article/pii/S0376042115300221.

[14] P. Huang, Y. Lu, M. Wang, Z. Meng, Y. Zhang, and F. Zhang. “Postcapture attitude takeover control of a partially failed spacecraft with parametric uncertainties”. In: *IEEE Transactions on Automation Science and Engineering* 16.2 (2018), pp. 919–930.

[15] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann. “Stable-Baselines3: Reliable Reinforcement Learning Implementations”. In: *Journal of Machine Learning Research* 22.268 (2021), pp. 1–8. url: https://jmlr.org/papers/v22/20-1364.html.

4

A Loss Landscape Visualization Framework for Interpreting Reinforcement Learning: An ADHDP Case Study

Reinforcement learning algorithms have been widely used in dynamic and control systems. However, interpreting their internal learning behavior remains a challenge. In the authors' previous work, a critic match loss landscape visualization method was proposed to study critic training. This study extends that method into a framework which provides a multi-perspective view of the learning dynamics, clarifying how value estimation, policy optimization, and temporal-difference (TD) signals interact during training. The proposed framework includes four complementary components: a three-dimensional reconstruction of the critic match loss surface that shows how TD targets shape the optimization geometry; an actor loss landscape under a frozen critic that reveals how the policy exploits that geometry; a trajectory combining time, Bellman error, and policy weights that indicates how updates move across the surface; and a state-TD map that identifies the state regions that drive those updates. The Action-Dependent Heuristic Dynamic Programming (ADHDP) algorithm for spacecraft attitude control is used as a case study. The framework is applied to compare several ADHDP variants and shows how training stabilizers and target updates change the optimization landscape and affect learning stability. Therefore, the proposed framework provides a systematic and interpretable tool for analyzing reinforcement learning behavior across algorithmic designs.

This chapter is based on the manuscript: J. Liu, J. Guo, and E. Gill, "A Loss Landscape Visualization Framework for Interpreting Reinforcement Learning: An ADHDP Case Study," submitted to *Acta Astronautica*, DOI: https://doi.org/10.48550/arXiv.2603.14600.

4.1. Introduction

Reinforcement learning (RL) algorithms have attracted researchers' attention in recent years. They have also been applied in dynamics and control problems.

Reinforcement learning control algorithms can be used in systems with uncertainties. For example, in an Active Debris Removal (ADR) mission, post-capture attitude control of the combined spacecraft encounters the challenges of system uncertainties, such as uncertain inertial parameters. In this case, reinforcement learning algorithms [1–3], which rely on the sampled data during the control process, rather than the model information, have also been used.

Although reinforcement learning has achieved success in various control tasks, it doesn't always perform as expected. RL algorithms are optimized within a training environment, which is a simulation of a specific physical system. The learned policies for one RL algorithm may not transfer effectively to new or slightly different systems [4]. In some cases, even a minor change in a system parameter can cause a previously well-tuned algorithm to fail [5]. RL's performance sensitivity to the training environment motivates the need to interpret the algorithm's performance. Understanding RL's training behavior is important for revealing the learning mechanism and thus improving performance robustness.

Recent studies have explored various approaches to interpret reinforcement learning behavior in control systems. For instance, some works visualize the state–action value function to reveal the learned control patterns [6], while others extract symbolic representations of the policy [7]. Methods such as sensitivity analysis, Jacobian-based visualization, and Koopman operator approximation [8] have also been introduced to relate RL policies to interpretable dynamical structures. In chapter 2, a visualization method that maps the critic match loss landscape was proposed to analyze the critic behavior during the training of actor–critic reinforcement learning algorithms. However, a single loss landscape alone cannot fully capture the interactions among the modules in the RL agent. Therefore, a more comprehensive and structured visualization framework is needed to support the interpretation of training dynamics and performance variations across reinforcement learning algorithms.

To help demonstrate the interpretation framework, one RL algorithm should be chosen as the sample. The Adaptive Dynamic Programming (ADP) method was proposed to obtain nearly optimal control strategies based on reinforcement learning (RL) and actor-critic (AC) architecture, which has received considerable attention in the past decades [9–13]. Among the ADP method, action-dependent heuristic dynamic programming (ADHDP), a form of Q-learning, has been been applied in post-capture control for the combined spacecraft [14, 15]. A notable characteristic of ADHDP is that it maintains a simpler structure, which allows faster training and adaptation in online implementation. On this basis, ADHDP is considered here to evaluate the potential of online reinforcement learning

methods for attitude control of rigid combined spacecraft with unknown inertia.

The ADHDP algorithm has shown its usage in the work mentioned above. Yet in practice, training instablities are often observed. Standard scalar metrics such as reward or TD error do not reveal exactly why training fails. These methods don't give enough information on the training information of the algorithm. Furthermore, when the ADHDP algorithm is equipped with deep learning techniques, such as a target neural network and different loss calculation methods, the underlying training procedure is not carefully studied. As a result, the ADHDP algorithm is used as a sample to explore how an interpretation framework can be applied to interpret the training process and control behavior of the ADHDP algorithm.

In this work, a visualization framework is proposed to interpret the training behavior of actor–critic reinforcement learning algorithms. The framework reshapes the critic match loss using the one-step temporal difference (TD) error, enabling the observation of critic convergence trends throughout training. To analyze the actor's adaptation process, an actor loss landscape is constructed to illustrate how policy updates respond to the critic's evaluation. A time–TD–actor weight plot is introduced to capture how the policy evolves with the change of Bellman error, while a state–TD plot identifies the state regions that contribute most to TD fluctuations. Together, these visualizations form an integrated framework for diagnosing the interaction between actor, critic, and TD error during reinforcement learning training. ADHDP variants with different deep learning techniques are selected as samples for the demonstration of the interpretation framework.

In this work, section 4.2 introduces the visualization framework and the ADHDP variants. section 4.3 presents the visualization results and analysis for ADHDP variants' performance using landscape and trajectory analyses. section 4.4 is the conclusion.

4.2. Method

In this section, the visualization framework for interpreting RL algorithms is introduced. The ADHDP algorithm is also given, which is used as an object to be interpreted using the visualization framework.

4.2.1. Visualizing the loss function

In reinforcement learning, neural networks are employed to approximate functions, using a large number of parameters. The loss landscape, describing how the loss changes with respect to these parameters, provides insights into characteristics such as flatness, sharpness, local minima, and saddle points. Yet, the high dimensionality of the parameter space renders direct visualization intractable. To overcome this issue, the Contour Plots and Random Directions method has been introduced [16]. This approach selects two directions, $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$, and evaluates the loss across the 2-D plane they span. The resulting

visualization corresponds to a projected slice of the overall loss landscape. The loss on this plane is defined as

$$f(\alpha,\beta) = \mathcal{L}(\boldsymbol{\zeta}^* + \alpha\boldsymbol{\delta} + \beta\boldsymbol{\eta}), \tag{4.1}$$

where $\boldsymbol{\zeta}^*$ denotes the reference point in the parameter space, and α and β are the step sizes along the two selected directions. The function $\mathcal{L}$ represents the loss value evaluated at each displaced position. This formulation transforms the high-dimensional optimization landscape into a three-dimensional surface, where $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$ define the two axes, (α,β) are the coordinates in this plane, and $f(\alpha,\beta)$ represents the vertical coordinate, thereby providing a clear and intuitive visualization of the loss landscape geometry.

4.2.2. Visualization technique framework

To give more insight into the performance of the ADHDP algorithm, more visualization figures are needed, besides the critic loss landscape visualization method. Here, the visualization technique framework based on subsection 4.2.1 is introduced.

To utilize the loss function visualization technique, four elements are needed for visualization, which are the recorded parameters, visualization indexes, batch data, and labels. The selection of these four elements will determine the meaning of the generated figures. In this work, four indices will be chosen to visualize the landscape, as shown in Table 4.1

Table 4.1: Summary of visualization indices in the proposed framework.

Visualization index	**Input data**	**Output figure**	**Interpretation focus**
Critic match loss landscape	Recorded critic weights; states-action samples and TD targets during training	3-D loss surface over critic weight space	Evolution of critic fitting behavior during training
Actor loss landscape	Recorded actor weights; frozen critic as cost function	3-D loss surface over actor weight space	Geometry and quality of the learned policy
Time–TD–actor weight trajectory	Actor weights and TD errors recorded across training steps	3-D trajectory of actor weight vs. TD vs. time	Coupling between policy drift and Bellman error
State–TD plot	System states and TD errors recorded during simulation	2-D plot of state vs. TD	State regions that dominate TD fluctuations

The four visualization indices involve multidimensional parameters that cannot be directly plotted. To address this, a general dimensionality-reduction method is recalled. Following the approach used in chapter 2, the parameters recorded during training are collected into a dataset, and Principal Component Analysis (PCA) is applied to extract the dominant directions of variation. The first one or two principal components directions, which correspond to $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$ in Equation 4.1, are then used to form a low-dimensional plane or axis for visualization. This allows the multidimensional parameter updates to be represented clearly before introducing the four specific visualization techniques. For each grid on the PCA plane, its corresponding weight is calculated using

$$\tilde{\mathbf{w}}_{\mathbf{c}}(\alpha,\beta) = \mathbf{w_c}^{r} + \alpha\boldsymbol{\delta} + \beta\boldsymbol{\eta}, \tag{4.2}$$

where $\mathbf{w_c}^{r}$ denotes the reference critic weight vector, $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$ are the two principal directions obtained from PCA of the recorded parameter trajectory. The loss value is evaluated on a grid over the coefficients (α,β).

The first index to be visualized is the critic match loss landscape. As discussed in chapter 2, the match loss landscape will show the trend of how the critic loss landscape changes during the training process. Same as the work in chapter 2, the critic weight at the end of each episode of training is recorded. The states-action samples and corresponding TD target recorded throughout training is selected for the calculation of the loss of each grid on the landscape.

$$L_{\text{match}}(\alpha,\beta) = \frac{1}{N_{\text{ref}}} \sum_{t=1}^{N_{\text{ref}}} \left(J(x_t^{\text{state}}, u_t^{\text{action}}; \tilde{\mathbf{w}}_c(\alpha,\beta)) - y_t^{\text{td}} \right)^2, \tag{4.3}$$

where $J(\cdot;\tilde{\mathbf{w}}_c(\alpha,\beta))$ denotes the critic network output under the reconstructed critic weight parameters, y_t^{td} is the TD target.

The second index to be visualized is the actor loss landscape. To fully study the training process of the RL algorithm, only showing the trend of how the critic loss changes is not enough. The actor is also approximated using neural networks. And it uses the information from the critic network as guidance. So the training of an actor neural network is also essential to evaluate the performance of the RL algorithm. With the critic frozen at the end of training as a surrogate for the cost $J(x,u)$, we define the actor loss over a fixed probe set of states $\mathcal{D}$ as

$$\mathcal{L}_{\text{actor}}(\boldsymbol{w}_a) = \frac{1}{|\mathcal{D}|} \sum_{x\in\mathcal{D}} \hat{J}(x,\ \pi_{\boldsymbol{w}_a}(x)), \tag{4.4}$$

where $\hat{J}(x,u)$ is the critic output and $\mathcal{D}$ is a fixed set of probe states collected during training.We record the actor parameters $\mathbf{w}_a$ at the end of each episode. To visualize the landscape, we evaluate $\mathcal{L}_{\text{actor}}(\partial)$ on a 2-D grid in parameter space constructed around the recorded trajectory, which is the PCA plane of $\{\mathbf{w}_a\}$, yielding an actor loss surface.

The third index to be visualized is the update of the actor weight with time and TD error. This index also shows how the actor's weight changes with TD error. The actor's weight is recorded after several training steps,

and the corresponding TD error is recorded. Since the actor's weight is a multi-dimensional parameter, the PCA technique is applied. The first main direction of the weight after PCA is selected for the visualization, together with time and TD error.

The last index to be visualized is the update of the state trajectory with TD error. Similar to how the third index is visualized, the states are recorded per several simulation steps, the corresponding TD error is recorded. PCA is also applied to the multidimensional states of the dynamic system. Then the states and TD error can be plotted in the 2-D figure. This visualization helps identify regions of the state space that contribute most to large TD errors.

4.2.3. ADHDP variants with deep learning techniques

To demonstrate how visualizing the critic loss landscape can aid in interpreting online RL algorithms, we take Action-dependent Heuristic Dynamic Programming (ADHDP) as a representative example. ADHDP is a particular reinforcement learning framework in which the control and learning objectives are separated and handled by two distinct neural networks: the actor and the critic, as illustrated in Figure 4.1 [17]. The critic network is optimized to approximate the total reward-to-go, serving as a surrogate for the cost function, while the actor network is trained so that the critic ultimately drives the system toward minimizing this cost.

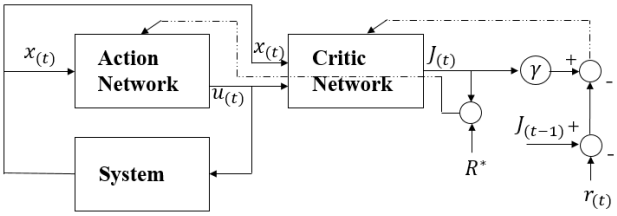


Figure 4.1: Structure of ADHDP

In ADHDP, the cost function $J(t)$ is formulated as

$$J(t) = r(t+1) + \gamma r(t+2) + \cdots = \sum_{k=1}^{\infty} \gamma^{k-1} r(t+k), \tag{4.5}$$

where $r(t+1)$ denotes the reinforcement signal (or reward) received by the system, and k indicates the time steps from instant t. The discount factor γ determines the relative importance of future rewards in the cumulative return.

The prediction error for the critic network, denoted by e_c, is defined as

$$e_c(t) = [r(t) + \gamma J(t)] - J(t-1), \tag{4.6}$$

which corresponds to the temporal-difference (TD) error commonly used in reinforcement learning.

Let $\mathbf{w}_c$ represent the weight parameters of the critic network. Their update follows the gradient descent principle. In this study, both the critic and actor networks are implemented as multilayer perceptrons (MLPs).

Figure 4.1 and Equation 4.6 describe the core of the ADHDP algorithm. The performance of the ADHDP algorithm will largely depend on the reinforcement learning techniques employed in the ADHDP algorithm. In this work, four versions of the ADHDP algorithm are interpreted using the visualization framework in subsection 4.2.2. The deep learning techniques employed by each version is shown in Table 4.2.

Table 4.2: DRL techniques applied in each ADHDP variant. The numbering corresponds to the order of variants discussed in the text.

No.	ADHDP variant	Target network	Numeric cost scaling	Huber critic loss	Target-policy smoothing (TPS)
1	Basic version	–	–	–	–
2	With target network	✓	–	–	–
3	With training stabilizers + TPS	✓	✓	✓	✓
4	With training stabilizers only	✓	✓	✓	–

The simplified version corresponds to the basic ADHDP algorithm described in chapter 2. The second version introduces a target network for the critic. The third version builds on the target-network version and further incorporates two training stabilizers, namely numeric cost scaling and the Huber critic loss. Target policy smoothing is also applied in this version. The fourth version removes the target policy smoothing used in the third version while retaining the two training stabilizers (Huber critic loss and numeric cost scaling).

Below, the DRL techniques employed in the ADHDP variants are introduced.

Target network Taking the critic network as an example, the target network is used to provide a less chaotic update to the critic network. A soft update ,the Polyak averaging, of the target network is applied in this work. After each gradient step on the online critic, the target critic is softly updated toward it:

$$\bar{\mathbf{w}}_c \leftarrow (1 - \tau_{\text{target}})\, \bar{\mathbf{w}}_c + \tau_{\text{target}}\, \mathbf{w}_c, \tag{4.7}$$

with a small $\tau_{\text{target}} \in (0, 1)$. Smaller τ_{target} yields a slower, more stable update.

Numeric cost scaling [18] This technique is defined as one of the two training stabilizers in this work. We rescale the per-step cost by a positive constant

scaling factor $\alpha_{\text{cost}} > 0$:

$$c'_t = \frac{1}{\alpha_{\text{cost}}}\, c_t, \qquad J'(x) = \frac{1}{\alpha_{\text{cost}}}\, J(x). \tag{4.8}$$

Here c_t denotes the original instantaneous cost at time step t, c'_t is the scaled cost used in training, $J(x)$ represents the value function associated with state x, and $J'(x)$ is the corresponding value function under the scaled cost.

Since this is a monotone linear transform, the optimal policy is unchanged. Only the numerical range of TD targets and gradients is compressed, which reduces loss curvature and gradient explosions.

Huber critic loss This technique is defined as one of the two training stabilizers in this work. Critic parameters $\mathbf{w}_c$ minimize a robust regression between prediction and target:

$$L_{\text{critic}} = \begin{cases} \frac{1}{2}\, e^2, & |e| \le \kappa_{\text{huber}}, \\ \kappa_{\text{huber}}|e| - \frac{1}{2}\kappa_{\text{huber}}^2, & |e| > \kappa_{\text{huber}}, \end{cases} \qquad e = \hat{J}_{\mathbf{w}_c}(x_t, u_t) - y_t^{\text{td}}, \tag{4.9}$$

where the symbols are defined as follows: y_t^{td} is the target value for the critic at time step t, typically obtained via bootstrapped estimates or TD targets. κ_{huber} is the threshold parameter in the Huber loss that determines the transition from quadratic (ℓ_2) to linear (ℓ_1) behavior. e is the prediction error of the critic, i.e., the difference between the predicted value $\hat{J}_{\mathbf{w}_c}(x_t, u_t)$ and the target y_t^{td}.

Near zero, the Huber loss behaves like the ℓ_2 loss, promoting smooth convergence, while in the tails it behaves like the ℓ_1 loss, down-weighting outliers arising from bootstrapping or distribution shifts.

Target-policy smoothing (TPS) [19] When forming the TD target, we evaluate the next action using a noisy, clipped version of the policy, then feed it to the target critic:

$$\tilde{u}_{t+1} = \text{clip}\left(\mu_\vartheta(x_{t+1}) + \epsilon_{\text{noise}},\, -u_{\max},\, u_{\max}\right), \qquad \epsilon_{\text{noise}} \sim \mathcal{N}(0, \sigma_{\text{noise}}^2 \mathbf{I}) \tag{4.10}$$

The TD target is

$$y_t^{\text{td}} = \begin{cases} c'_t + \gamma\, \hat{J}_{\bar{\varphi}}(x_{t+1}, \tilde{u}_{t+1}), & \text{if not terminal,} \\ c'_t + \text{penalty}', & \text{terminal.} \end{cases} \tag{4.11}$$

Here $\hat{J}_{\bar{\varphi}}$ denotes the slow-moving target critic network obtained via soft updates. The term ϵ_{noise} represents the policy smoothing noise added to the next action, which is sampled from a zero-mean Gaussian distribution with standard deviation σ_{noise}. The parameter σ_{noise} controls the magnitude of the injected noise. $\mathbf{I}$ denotes the identity matrix, ensuring that the noise is applied independently to each action dimension. c'_t is the scaled instantaneous cost

used for critic training, obtained from the original cost c_t through the numeric cost scaling described previously. The clipping operation enforces the actuator bounds defined by the maximum control torque $u_{\max}$.

TPS acts as a local averaging operator over the action dimension, reducing over-estimation bias and spiky curvature in $J(x, u)$ around the greedy action.

4.3. Results for demonstration of the framework

In this section, the spacecraft attitude system is introduced. The control results of using different versions of the ADHDP algorithm are shown. The control results will be interpreted using the loss landscape visualization framework. Raw loss is used in this section to preserve the original landscape geometry for training behavior interpretation.

4.3.1. Attitude dynamics of spacecraft with unknown inertia

In this study, the combined spacecraft in the ADR scenario is taken as the control system for testing ADHDP variants. As reviewed in section 4.1, the combined spacecraft in an ADR mission is subject to multiple sources of uncertainty. To evaluate the control algorithm for the post-capture phase in a progressive manner, we begin with the simplest case, where uncertainty arises solely from the inertial parameters. The following assumptions are made:

- The target is firmly grasped by rigid robotic manipulators mounted on a rigid servicing spacecraft.
- After capture, all manipulator joints are locked [20].
- The target is rigid, uncooperative, and has no control capability.

Under these conditions, the combined spacecraft can be modeled as a single rigid body with unknown inertial properties.

In this work, the reinforcement learning algorithm is trained in simulation instead of on the physical spacecraft. The training process produces an initial stabilizing control policy represented by the actor network. During operation, the learned policy is used as a feedback controller that generates control inputs from the observed spacecraft states. Training in simulation improves safety, since the real system does not need to start from an untrained policy. In addition, the visualization framework can be used during training to examine the evolution of the critic and actor optimization, providing insight into the learning process before deployment.

A body-fixed reference frame $\mathcal{B}$ is defined at the center of mass of the combined spacecraft, with its axes aligned to the principal inertia directions. An Earth-Centered Inertial (ECI) frame $\mathcal{N}$ is also introduced, spanned by the unit vectors $\{\overline{\boldsymbol{n_1}}, \overline{\boldsymbol{n_2}}, \overline{\boldsymbol{n_3}}\}$. The $\overline{\boldsymbol{n_1}}$ axis points toward the mean equinox, $\overline{\boldsymbol{n_3}}$ is aligned with the Earth's rotation axis (celestial north), and $\overline{\boldsymbol{n_2}}$ completes the right-handed triad.

The spacecraft attitude is represented by a unit quaternion

$$\boldsymbol{q} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix},$$

which specifies the rotation from $\mathcal{N}$ to $\mathcal{B}$. This formulation avoids singularities and is well-suited for large-angle maneuvers. The angular velocity, expressed in frame $\mathcal{B}$, is denoted as

$$\boldsymbol{\omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}.$$

The kinematic equation of motion is

$$\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}, \tag{4.12}$$

where $\otimes$ denotes quaternion multiplication.

The attitude dynamics are governed by

$$\boldsymbol{M} = \hat{J}_{\text{sc}} \cdot \dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \left(\hat{J}_{\text{sc}} \cdot \boldsymbol{\omega}\right), \tag{4.13}$$

where $\boldsymbol{M}$ is the applied control torque, and $\hat{J}_{\text{sc}}$ is the (unknown) inertia matrix of the combined spacecraft.

4.3.2. Common simulation setup

The following settings are shared across all ADHDP simulations. The integration step is set to $dt = 0.01\,\text{s}$, with 5000 steps per episode and 100 episodes in total.

Regarding the setup of the inertia parameters of the spacecraft, in practical ADR scenarios, the inertia properties of the combined spacecraft are generally uncertain due to the unknown mass distribution of the captured target. In this work, the control problem is therefore formulated as the stabilization of a rigid spacecraft with unknown inertia parameters. The reinforcement learning algorithms considered in this study are model-free, meaning that the controller does not require knowledge of the inertia matrix or other system parameters. Instead, the control policy is learned through interaction with the system dynamics. For simulation purposes, however, the spacecraft dynamics must be generated using a specific inertia matrix. Therefore, a fixed inertia value is used in the numerical simulations to represent the dynamics of the combined spacecraft. This inertia value is not provided to the learning algorithm and is only used within the simulation environment to produce state transitions. The spacecraft inertia matrix is

$$J = \begin{bmatrix} 1.0 & 0.02 & 0.02 \\ 0.02 & 0.8 & 0.03 \\ 0.02 & 0.03 & 0.9 \end{bmatrix},$$

and no external disturbance torque is applied. The maximum control torque is limited to $0.5\,\mathrm{N{\cdot}m}$. The initial attitude corresponds to a 20° rotation about the axis $[1.0, -0.5, 0.2]$, with zero initial angular velocity. The discount factor is $\gamma = 0.95$. Both actor and critic are two-layer MLPs with 64 hidden units, trained with learning rates of 1×10^{-4} and 1×10^{-3}, respectively. Gaussian policy noise decays from 0.02 to 0.005 over $50{,}000$ steps. The cost are defined using Equation 4.14,

$$c_t = k_{\text{att}}\left(1 - q_0^2\right) + k_{\text{rate}}\,\|\boldsymbol{\omega}\|^2 + k_{\text{torque}}\,\|\boldsymbol{M}\|^2, \tag{4.14}$$

with weights coefficients $k_{\text{att}} = 20.0$, $k_{\text{rate}} = 2.0$, and $k_{\text{torque}} = 0.1$. Gradients are clipped at 1.0, and a termination penalty of 300 is applied if the attitude error exceeds $2.8\,\mathrm{rad}$ or the angular velocity norm exceeds $8\,\mathrm{rad/s}$.

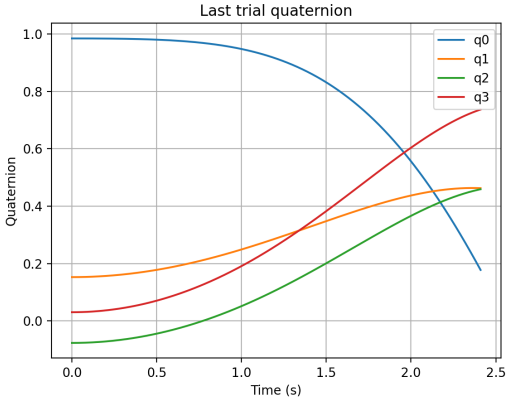
4.3.3. Basic ADHDP

Under the given common simulation setup above, the simulation results for spacecraft attitude control using the basic ADHDP algorithm are shown below.

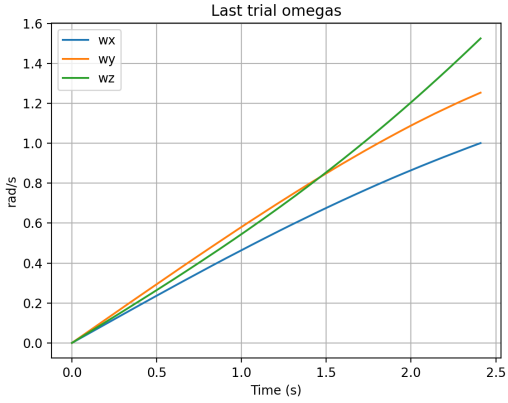
With the basic version of the ADHDP algorithm, Figure 4.2 shows the quaternions, angular velocities, and control torque of the system during the last episode of training. It shows the failed control result. The torques of the two axes are almost saturated. With the figures below, the reasons for failure will be given.

Figure 4.3 shows the loss surface of the basic version of ADHDP during training. From Figure 4.6a we can see that the critic match loss is sharp in one direction yet quite flat in another direction. Figure 4.3b shows the actor loss landscape under the final critic network. The surface is flat in the first direction, yet it shows a much smaller range of change in the second direction. It indicates that, under the current critic guidance, a large region of actions gives equal solutions. This flat valley makes the policy gradient easy to stop near saturation.

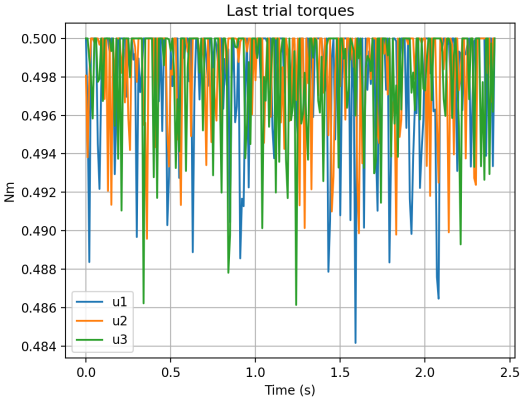
Figure 4.4b shows the mean TD of each episode during training. The TD error first rapidly increases and then slowly decreases, and finally oscillates around 1.3. It shows no convergence trend. Figure 4.4b shows the first principle of the actor weight change with time and TD error. It has a variance of 97.2%, indicating that the weight evolution is almost in a single direction. The curve increases rapidly in the front part and slows down in the middle and back parts. It is consistent with the actor surface in Figure 4.3b. The policy drifts in large steps in the early stage and is limited by the flat valley in the later stage. Figure 4.4c shows how the first principle direction of states changes with time and TD. It has an extremely high TD spike at the end of the state, while the rest of the graph is relatively low. Most states exhibit relatively small TD errors, indicating that the critic approximates the value function reasonably well in the commonly visited regions of the state space. However, the extreme PC1 region shows a sharp TD spike, suggesting that the critic fails to generalize to states with large attitude errors and angular velocities. The distribution of states



(a) Quaternions

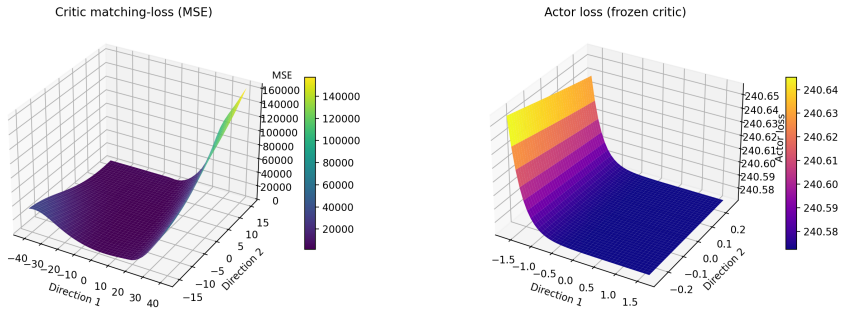


(b) Angular velocity



(c) Control Torque

Figure 4.2: State trajectory, basic ADHDP



(a) Critic Match Loss Landscape (b) Actor Loss Landscape

Figure 4.3: Loss surface, basic ADHDP

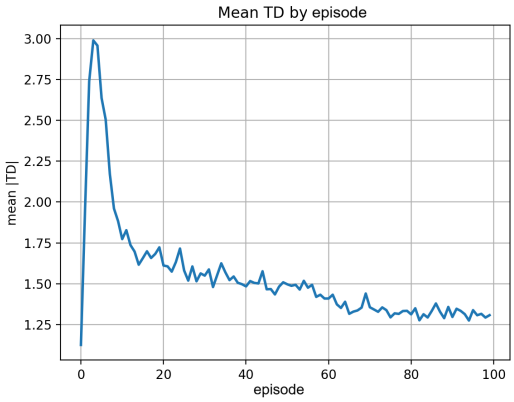
differs significantly from that of the early training phase, making it difficult for the critic to generalize.

From the figures above, we can conclude that the failure of this simulation is caused by the unstable update of the critic net, which leads the actor net to saturate at the boundary of the flat valley. These two factors together lead to the divergence of training.

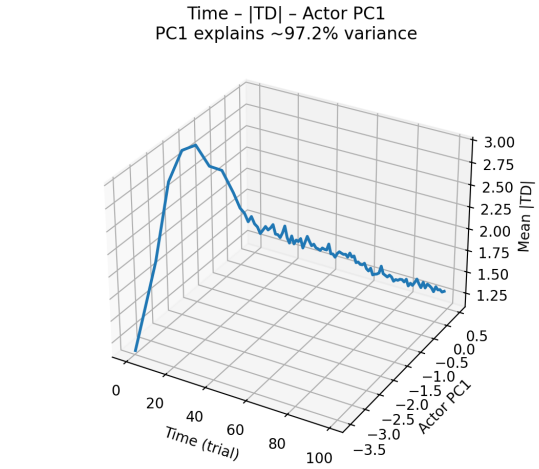
The above analysis indicates that the failure of the basic ADHDP algorithm is closely related to the instability of the critic update. The TD error exhibits significant fluctuations during training, and the critic loss landscape shows strong anisotropy, suggesting that the critic is trained with rapidly changing targets. Since the actor update directly depends on the critic estimates, this instability propagates to the actor network and leads to saturation in the flat valley of the actor loss landscape. To improve the stability of the critic update, a target network can be introduced in the ADHDP framework. In the following section, the same diagnostic framework is applied to analyze the training behavior of ADHDP with a target network and to examine how the target network influences the critic and actor dynamics.

4.3.4. ADHDP with target network

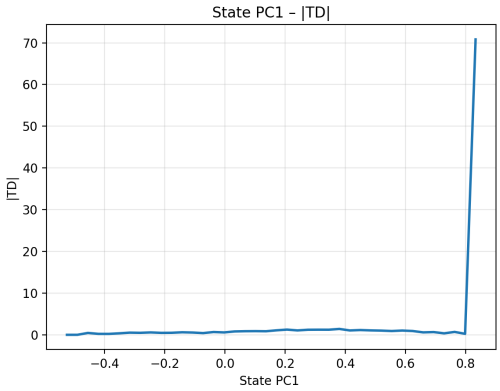
This variant of the ADHDP algorithm incorporates a target critic network with Polyak averaging with $\tau_{\text{target}} = 0.005$ to stabilize the critic bootstrap. The actor and critic networks are otherwise configured as in the common setup. Training uses a delayed actor update. The actor is updated every 5 steps, starting after 30000 steps of pretraining freeze. Pretrain freeze is applied in training. Actor parameters are frozen for the first 10,000 environment steps. Critic trains normally during this phase. After pretrain freeze, actor is updated only every 10 steps. Gradient clipping with a maximum norm of 1.0 and L2 regularization $\lambda_{\text{actor}} = 0.1$ are applied. Exploration noise is applied to the action. Environment



(a) TD by episode



(b) Actor weight with time and TD



(c) State with TD

Figure 4.4: TD trajectory, basic ADHDP

action noise decays from $\sigma_{\text{noise,init}} = 0.05$ to $\sigma_{\text{noise, final}} = 0.02$ over 50,000 steps.

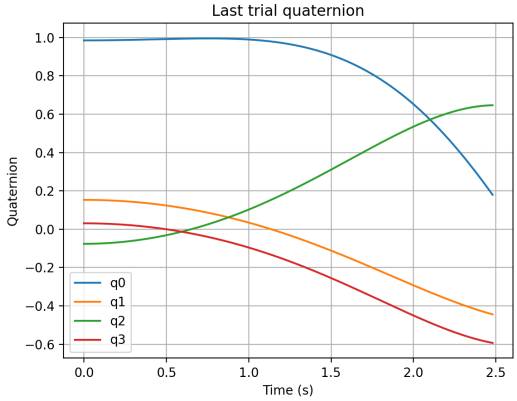
With the version of the ADHDP algorithm with target net, Figure 4.5 shows the quaternions, angular velocities, and control torque of the system during the last episode of training. Quaternions and attitude angular velocity trajectories are still unstable. Control toques still have obvious saturation. The reasons for the non-convergence will be given.

In Figure 4.6a, it shows that the critic match loss approximates a wide and shallow bowl, with an upward trend in the front right. In the upper right region of the critic match loss landscape, small changes in the critic weight can lead to large changes in the critic match error. In Figure 4.6, it shows that the actor loss surface exhibits a typical wedge-shaped valley, with a steep slope along Direction-1 and a relatively flat slope along Direction-2. With a frozen critic, the actor loss exhibits a strong gradient along one direction but remains nearly flat along another direction. Under the action cap constraint, the weak gradient region is easily trapped at the saturation boundary, resulting in near-saturation of the control torque.

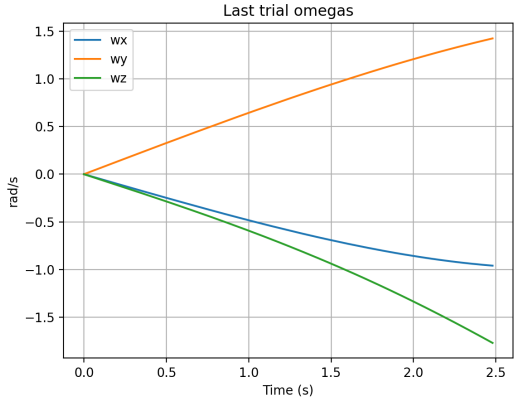
In Figure 4.7a, it shows that the TD error increases rapidly in the early stages, then slowly decreases to a value that is still high in the middle stages. Compared to the basic ADHDP version, the TD error fluctuates within a narrower range throughout the entire process. This indicates that using the target net can mitigate jitter, but it cannot reduce abnormal TD error to a sufficiently small value to allow the states to converge to a stable value. In Figure 4.7c, it shows that the states have a huge spike at the end, indicating that the TD error becomes extremely large when the system approaches unstable regions of the state space.

From the visualization framework, the reason for the control failure can be further explained. The critic match loss landscape in Figure 4.6a becomes smoother in the loss compared with the basic ADHDP version, which shows that the target network stabilizes the critic fitting process. However, the surface still contains a long ridge region where the loss changes rapidly along one direction. When the actor updates its parameters under this critic, the policy gradient mainly follows the steep direction of the actor loss surface while remaining weak along the flat direction, as shown in Figure 4.6. Under the action cap constraint, this imbalance easily drives the policy toward the saturation boundary. As a result, the control torques remain close to their limits, which can be observed in Figure 4.5. Once the system enters these saturated control regions, the collected samples contain large TD errors, which further correspond to the spike observed in the state–TD plot in Figure 4.7c. Therefore, although the target network makes the TD curve smoother, the geometry of the critic and actor landscapes still leads the policy to saturated actions, and the closed-loop control eventually becomes unstable.

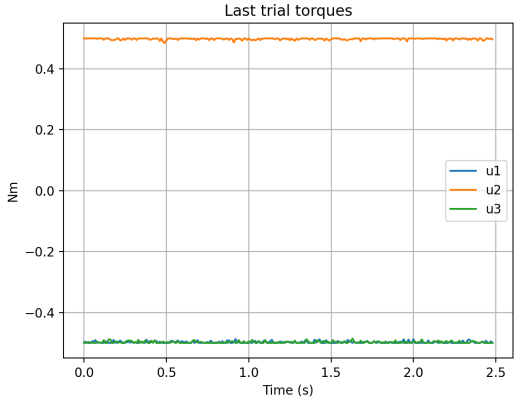
This consistency among the visualizations shows that the target network mainly slows down the update of TD targets. It helps to reduce short-term fluctuation but does not change the overall shape of the loss landscape or the distribution of the learning signals. From the visualization framework, it can be



(a) Quaternions

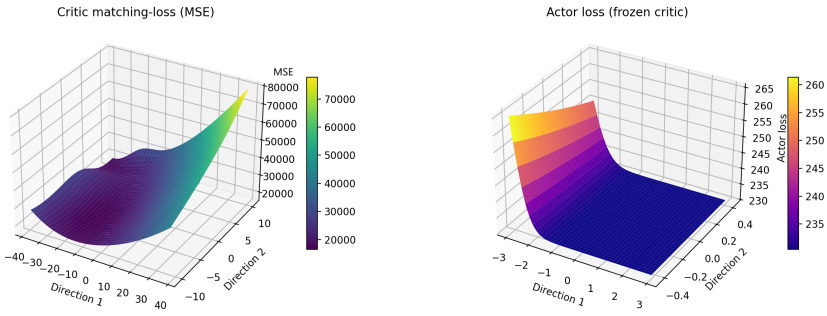


(b) Angular velocity



(c) Control Torque

Figure 4.5: State trajectory, ADHDP with target net



(a) Critic Match Loss Landscape (b) Actor Loss Landscape

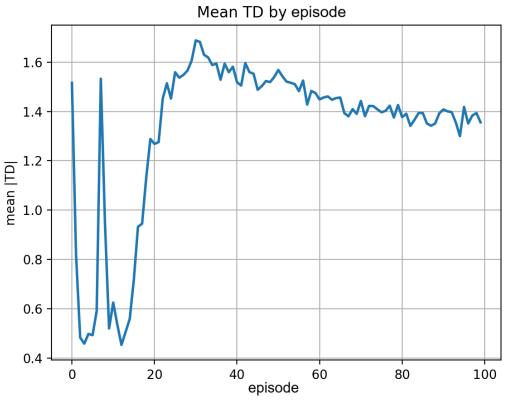
Figure 4.6: Loss surface, ADHDP with target net

seen that the target network makes the training more stable, but it does not improve the convergence of the algorithm.

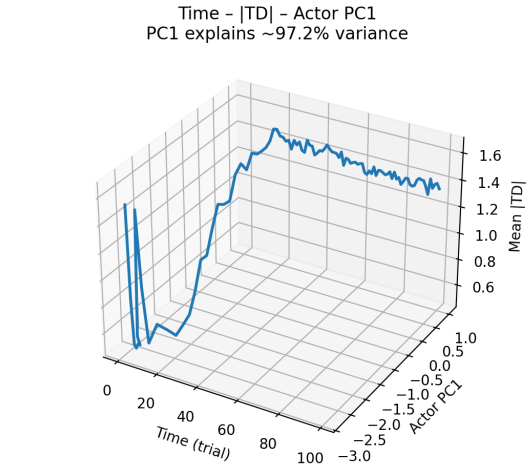
Although the target network improves the stability of the critic update, the previous analysis shows that it does not fundamentally resolve the unfavorable geometry of the loss landscapes. The critic match loss surface still contains elongated ridge regions, and the actor loss surface exhibits a strong imbalance between steep and flat directions. Under the action cap constraint, this imbalance easily drives the policy toward saturation, which eventually leads to unstable closed-loop behavior. To further improve the stability of the training process, additional stabilization techniques are introduced in the next version of ADHDP. Building upon the target-network framework, this version incorporates two training stabilizers, namely numeric cost scaling and the Huber critic loss. In addition, target policy smoothing is applied to reduce the sensitivity of the critic to small variations in the policy. The following analysis applies the same visualization framework to examine how these mechanisms influence the critic and actor training dynamics.

4.3.5. ADHDP WITH TRAINING STABILIZERS AND TARGET POLICY SMOOTHING

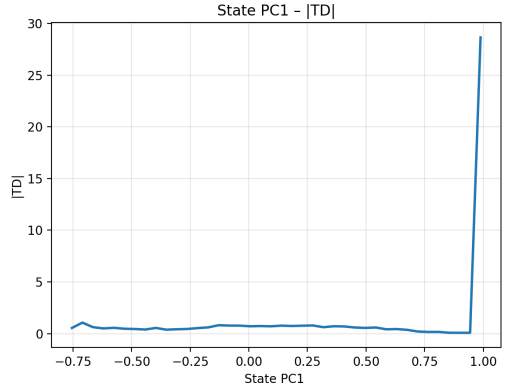
Compared with the ADHDP with target net, this variant introduces several stabilizing and training techniques. It shares the same simulation set up as the ADHDP with target net, but it has two more stablization techniques and target policy smoothing. Huber loss for critic is used in this version. Critic loss uses smooth ℓ_1 Huber loss instead of standard MSE. Cost scaling is also used, which means that all immediate costs are scaled by factor 10.0 to improve numerical stability without changing the optimal point.Target action smoothing is also added. Additive noise on the target action used in TD bootstrap. Smoothing noise standard deviation $\sigma_{\text{noise, smooth}}$ is set to 5% of the torque limit used in the controller..



(a) TD by episode

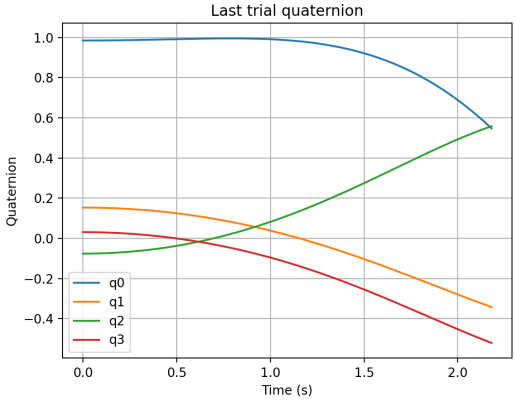


(b) Actor weight with time and TD

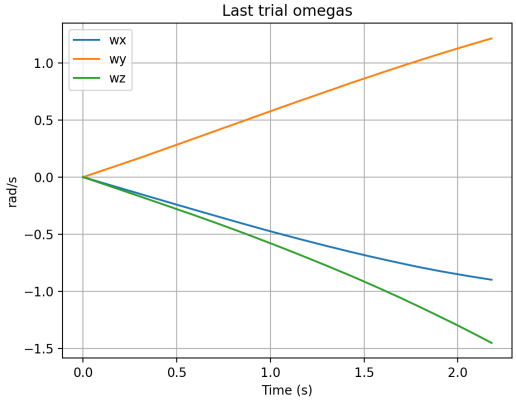


(c) State with TD

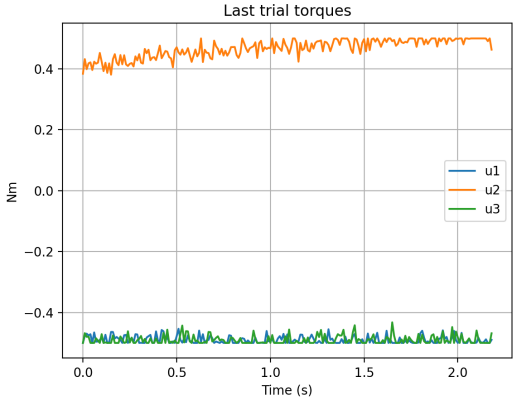
Figure 4.7: TD trajectory, ADHDP with target net



(a) Quaternions



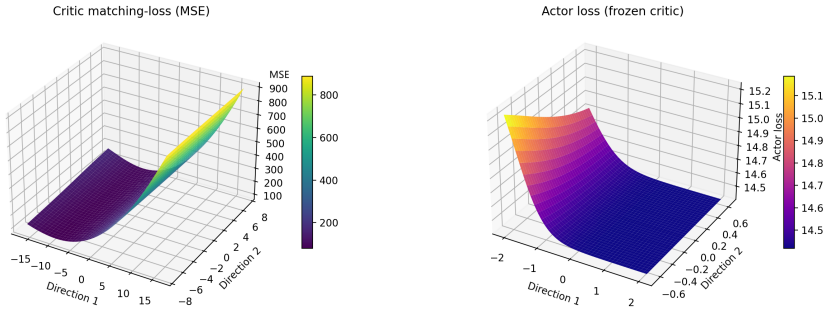
(b) Angular velocity



(c) Control Torque

Figure 4.8: ADHDP with training stabilizers (Huber critic loss and numeric cost scaling) and target policy smoothing

With the version of the ADHDP algorithm training stabilizers and target policy smoothing, Figure 4.8 shows the quaternions, angular velocities, and control torque of the system during the last episode of training. It shows the failed control result. With the figures below, the reasons for failure will be given.



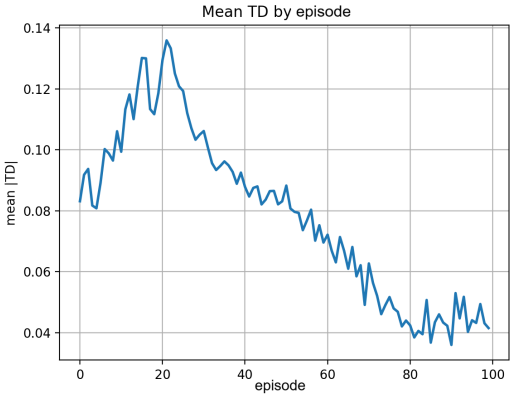
(a) Critic Match Loss Landscape (b) Actor Loss Landscape

Figure 4.9: Loss surface, ADHDP with training stabilizers (Huber critic loss and numeric cost scaling) and target policy smoothing

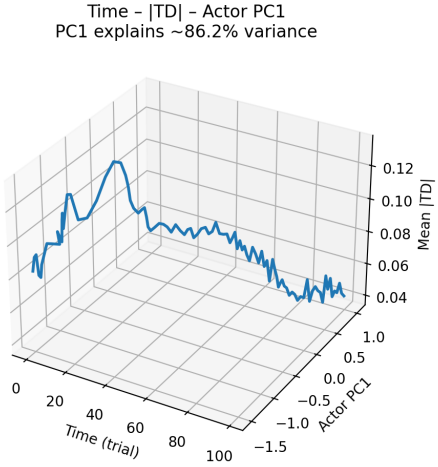
In Figure 4.9a, it shows that the critic match loss landscape is more rounded and blunt compared to the ADHDP with the target net version, with no obvious long ridges. This indicates that the introduction of the Huber loss formulation and cost scaling truncates anomalous TD gradients. Together with target policy smoothing, the gradients in the critic loss landscape become more gentle, making it easier to learn. In Figure 4.9, it shows that the actor loss landscape still exhibits a wedge shape, but with a smoother valley. This is consistent with the smoother and rounder nature of the critic match loss landscape.

In Figure 4.10a, it shows that the TD variation is smaller than that of the basic ADHDP version and the ADHDP with target net, gradually converging to smaller values. In Figure 4.10b, it shows that the first direction of actor weight exhibits a monotonically drifting trend while TD decreases during training. The PCA variance decreases from 97%, which is the variance in the target ADHDP version, to 86%, indicating that the weight evolution is still dominated by one main direction. In Figure 4.10c, medium-sized peaks and a flatter surface are shown compared to the basic ADHDP version and the ADHDP with target net version, indicating that extreme endpoints are suppressed by the Huber loss and cost scaling.

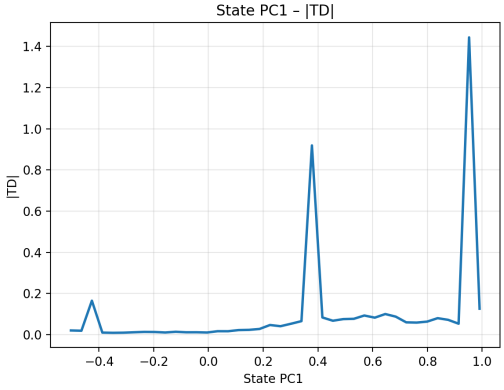
However, the visualization framework also explains why the control still fails. Although the critic loss landscape in Figure 4.9a becomes smoother and more rounded after introducing Huber loss, cost scaling, and target policy smoothing, the global geometry of the landscape still contains a dominant slope along Direction-1. As a result, the actor loss surface in Figure 4.9b still exhibits a wedge-shaped valley, although the valley becomes smoother. This means that



(a) TD by episode



(b) Actor weight with time and TD



(c) State with TD

Figure 4.10: TD trajectory, ADHDP with training stabilizers (Huber critic loss and numeric cost scaling) and target policy smoothing

the policy gradient remains much stronger in one direction than in the other. During training, the actor parameters therefore drift mainly along this dominant direction, which is consistent with the trajectory shown in Figure 4.10b. Although the TD magnitude decreases during training, the policy update is still biased toward this direction and cannot sufficiently explore other directions of the policy space. Under the torque saturation constraint, this biased update easily pushes the policy toward the saturation boundary. Consequently, the control torques remain close to their limits, which can be observed in Figure 4.8. Once the control inputs approach saturation, the spacecraft states cannot be corrected effectively and the attitude error gradually accumulates, eventually leading to divergence of the closed-loop control. This also explains why the TD curves appear smoother while the system trajectories in Figure 4.8 still fail to converge.

In conclusion, the Huber loss truncates the gradients of abnormal TD, significantly reducing the range of TD error and the peaks in state-TD. The cost scaling technique avoids excessive fluctuations in the loss landscape, making it easier to train the actor and critic weights. Target policy smoothing makes the critic loss landscape more rounded and the actor loss landscape more flat, allowing the TD error to decrease more smoothly during training. The combined effect of multiple stabilizers improves training metrics. However, these improvements do not fundamentally change the anisotropic geometry of the actor loss landscape, which still drives the policy toward saturated actions.

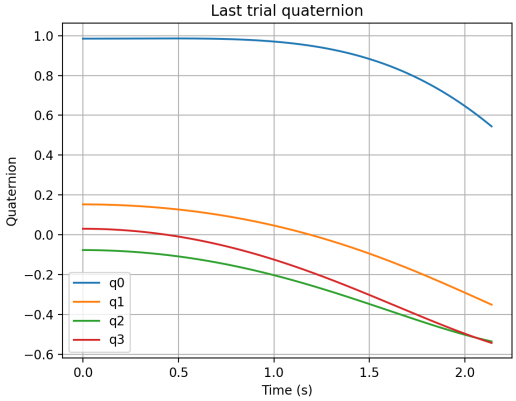
Although the combination of training stabilizers and target policy smoothing improves the smoothness of the loss landscapes and reduces the TD error magnitude, the closed-loop control still fails to converge. Since several stabilization mechanisms are introduced simultaneously in the previous version, it is important to further examine the specific role of target policy smoothing in the training dynamics. Therefore, in the next experiment, the target policy smoothing mechanism is removed while retaining the two training stabilizers, namely the Huber critic loss and cost scaling. Using the same visualization framework, the resulting changes in the critic loss landscape, actor optimization trajectory, and TD distribution are analyzed.

4.3.6. ADHDP with training stabilizers

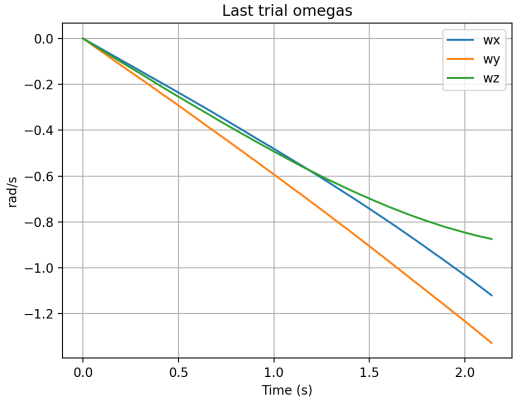
In this version, the target policy smoothing is removed compared with the version of ADHDP with training stabilizers and target policy smoothing. The control results and visualization results are shown in the following figures.

With the version of the ADHDP algorithm training stabilizers, Figure 4.11 shows the quaternions, angular velocities, and control torque of the system during the last episode of training. It shows the failed control result. With the figures below, the reasons for failure will be given.

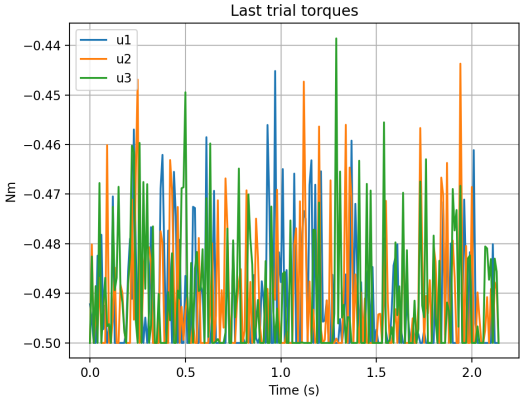
In Figure 4.12a, it shows that the critic match loss landscape exhibits a very sharp and narrow peak, much sharper than the ADHDP version with training stabilizer and target smoothing. This is due to the sharp terrain induced by the



(a) Quaternions

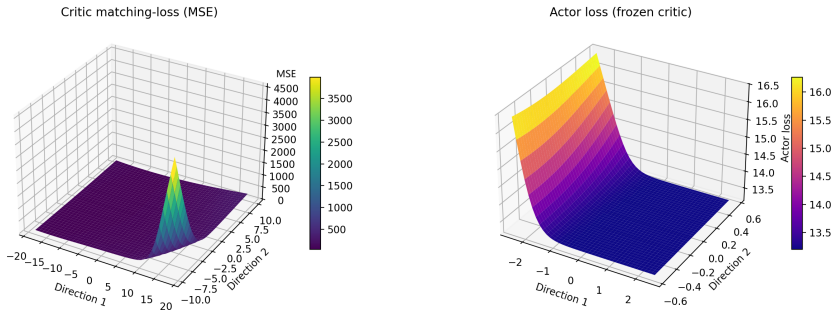


(b) Angular velocity



(c) Control Torque

Figure 4.11: State trajectory, ADHDP with training stabilizers(Huber critic loss and numeric cost scaling), no target policy smoothing



(a) Critic Match Loss Landscape (b) Actor Loss Landscape

Figure 4.12: Loss surface, ADHDP with training stabilizers(Huber critic loss and numeric cost scaling), no target policy smoothing

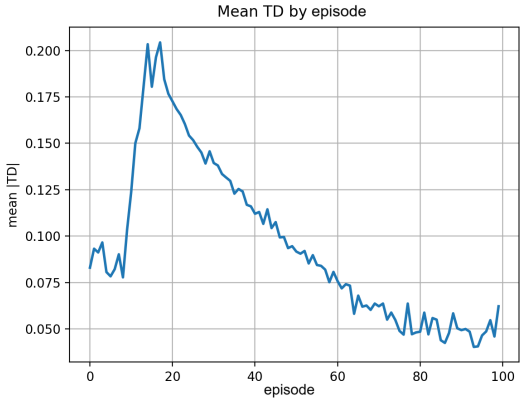
deterministic target without target policy smoothing. The actor loss landscape still exhibits a wedge shape.

In Figure 4.13a, it shows that the TD error varies within a relatively narrow range and gradually decreases to a lower value during training. This indicates that the Huber loss and cost scaling still help suppress large TD fluctuations. However, the loss landscapes reveal that this decrease of TD does not correspond to a good optimization region. As shown in Figure 4.12a, the critic match loss landscape exhibits a very sharp and narrow peak, which is much steeper than the version with target policy smoothing. This sharp geometry is caused by the deterministic TD target without smoothing. The actor loss landscape in Figure 4.12b still exhibits a wedge-shaped valley, indicating that the policy gradient remains much stronger along one direction.

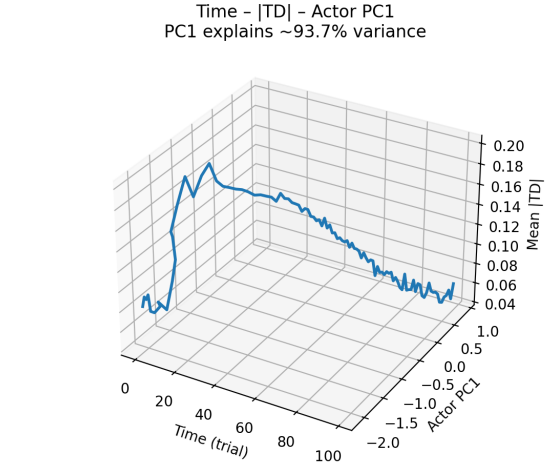
This behavior is also reflected in the training trajectory. In Figure 4.13b, the TD decreases while the first principal direction of the actor weight gradually moves toward a single dominant direction. This shows that the actor update is still biased and does not sufficiently explore other directions of the policy space. The state–TD relation in Figure 4.13c still contains two clear peaks. Although the amplitude of these peaks is smaller than in the basic ADHDP and target network versions, they indicate that certain state regions still produce relatively large TD errors.

Under this biased update direction, the control policy is gradually pushed toward the torque saturation boundary. As shown in Figure 4.11, the control torques remain close to the saturation limit during the last trial. Once the control inputs approach the limit, the controller cannot generate sufficient corrective torque to stabilize the spacecraft. As a result, the angular velocity and quaternion trajectories fail to converge. These observations explain why the TD curve appears smooth while the closed-loop control still diverges.

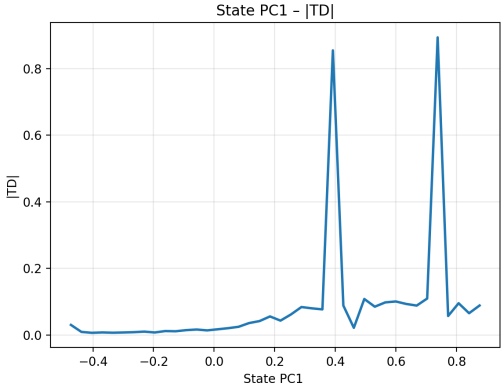
In conclusion, after removing target policy smoothing, the critic match loss



(a) TD by episode



(b) Actor weight with time and TD



(c) State with TD

Figure 4.13: TD trajectory, ADHDP with training stabilizers(Huber critic loss and numeric cost scaling), no target policy smoothing

returns to deterministic behavior, causing the spike in the critic match loss landscape to appear again. However, due to the presence of Huber, cost scaling, and small noise, the overall TD trend continues to decline. However, the policy updates guided by this are very stiff, causing the control torque to be trapped in a near-saturation state. These results clarify the role of target policy smoothing in the ADHDP framework. While the Huber loss and cost scaling mainly reduce TD magnitude and suppress extreme TD spikes, target policy smoothing primarily influences the geometry of the critic loss landscape by preventing the formation of sharp peaks. However, even with these stabilizers, the actor optimization remains strongly biased toward a dominant direction, which ultimately leads to saturation of the control torque and failure of the closed-loop control.

A comparison across the four ADHDP variants reveals a consistent mechanism underlying the failure of the closed-loop control. Although the introduction of target networks, Huber loss, cost scaling, and target policy smoothing significantly stabilizes critic learning and reduces TD fluctuations, these improvements do not fundamentally change the optimization geometry of the actor. In all variants, the actor loss landscape consistently exhibits a strongly anisotropic wedge-shaped structure, where the policy gradient is dominant along one direction while remaining weak along others. Under the torque saturation constraint, this geometry gradually drives the policy toward the action boundary. Once the control torques approach saturation, the controller loses the ability to generate effective corrective torques, and the spacecraft states eventually diverge. These observations suggest that the primary difficulty of the system does not lie solely in critic instability, but rather in the unfavorable optimization geometry arising from the coupling between actor and critic updates. Improvements in critic stability can smooth the TD signal and reduce short-term fluctuations, yet they do not fundamentally alter the dominant-direction structure of the actor loss landscape. Consequently, the learned policy still tends to drift toward the saturation boundary, ultimately leading to control failure.

The visualization results therefore suggest that further improvements should focus not only on stabilizing critic learning, but also on improving the geometry of the actor optimization landscape. Possible directions include introducing stronger regularization on control effort, adjusting the actor output scaling relative to the torque limits, or designing objective formulations that penalize near-saturation actions more explicitly. Such approaches may help reshape the actor loss landscape and prevent the policy from being trapped near the saturation boundary while preserving the lightweight online structure of the ADHDP framework.

From the results discussion above, the following summary can be drawn. The visualization framework provides a clear view of how different parts of the actor–critic algorithm evolve during training. The critic match loss landscape shows how the critic fits the TD targets and whether the critic optimization reaches a stable region in the parameter space. The actor loss

landscape reveals the shape of the policy and whether the actor update follows a smooth direction. The time–TD–actor weight trajectory illustrates how the actor parameters change together with the TD signal, while the state–TD plot identifies which system states produce large TD errors. Together, these visualizations explain how the critic, actor, and system states interact during learning, and how their coupling influences training stability and control performance. Through these observations, the influence of different ADHDP modifications can be directly identified, demonstrating that the framework is useful for analyzing and comparing reinforcement learning algorithms. This indicates that the framework provides a practical tool for interpreting the training behavior of actor–critic reinforcement learning algorithms.

4.4. Conclusion

In this work, a loss landscape visualization framework is developed to interpret the training process of actor–critic reinforcement learning algorithms. The framework includes four visualization components. The critic loss landscape shows how the critic fits the TD targets during training. The actor loss landscape, with a frozen critic, reflects the shape and quality of the learned policy. The state–TD plot shows which state regions produce large TD errors, and the actor weight–time–TD plot tracks how the policy changes with the TD error. Four ADHDP variants are used to demonstrate the framework. The results show how training stabilizers and target updates change the loss landscape and affect training stability. The visualizations also show that a small TD error does not always mean a good policy. Overall, the proposed framework provides a systematic approach for diagnosing reinforcement learning algorithms and for understanding how algorithmic design choices influence training behavior and control performance.

References

[1] C. E. Oestreich, R. Linares, and R. Gondhalekar. “Autonomous six-degree-of-freedom spacecraft docking with rotating targets via reinforcement learning”. In: *Journal of Aerospace Information Systems* 18.7 (2021), pp. 417–428.

[2] M. Tipaldi, R. Iervolino, and P. R. Massenio. “Reinforcement learning in spacecraft control applications: Advances, prospects, and challenges”. In: *Annual Reviews in Control* 54 (2022), pp. 1–23.

[3] S. Rafiee, M. Kankashvar, P. Mohammadi, and H. Bolandi. “Active fault-tolerant attitude control based on Q-learning for rigid spacecraft with actuator faults”. In: *Advances in Space Research* 74.3 (2024), pp. 1261–1275.

[4] C. Packer, K. Gao, J. Kos, P. Krähenbühl, V. Koltun, and D. Song. “Assessing generalization in deep reinforcement learning”. In: *arXiv preprint* (2018). arXiv:1810.12282.

[5] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. “Sim-to-real transfer of robotic control with dynamics randomization”. In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 3803–3810.

[6] C. Glanois, P. Weng, M. Zimmer, D. Li, T. Yang, J. Hao, and W. Liu. “A survey on interpretable reinforcement learning”. In: *Machine Learning* 113.8 (2024), pp. 5847–5890.

[7] D. Hein, S. Limmer, and T. A. Runkler. “Interpretable control by reinforcement learning”. In: *IFAC-PapersOnLine* 53.2 (2020), pp. 8082–8089.

[8] P. Rozwood, E. Mehrez, L. Paehler, W. Sun, and S. L. Brunton. “Koopman-assisted reinforcement learning”. In: *arXiv preprint arXiv:2403.02290* (2024).

[9] F. L. Lewis and D. Vrabie. “Reinforcement learning and adaptive dynamic programming for feedback control”. In: *IEEE circuits and systems magazine* 9.3 (2009), pp. 32–50.

[10] Y. Ouyang, L. Dong, Y. Wei, and C. Sun. “Neural network based tracking control for an elastic joint robot with input constraint via actor-critic design”. In: *Neurocomputing* 409 (2020), pp. 286–295.

[11] Z. Zheng, L. Ruan, M. Zhu, and X. Guo. “Reinforcement learning control for underactuated surface vessel with output error constraints and uncertainties”. In: *Neurocomputing* 399 (2020), pp. 479–490.

[12] B. Sun and E.-J. van Kampen. “Reinforcement-learning-based adaptive optimal flight control with output feedback and input constraints”. In: *Journal of Guidance, Control, and Dynamics* 44.9 (2021), pp. 1685–1691.

[13] Y. Zhou, E.-J. Van Kampen, and Q. Chu. “Incremental model based online heuristic dynamic programming for nonlinear adaptive tracking control with partial observability”. In: *Aerospace Science and Technology* 105 (2020), p. 106013.

[14] C. Wei, J. Luo, H. Dai, Z. Bian, and J. Yuan. “Learning-based adaptive prescribed performance control of postcapture space robot-target combination without inertia identifications”. In: *Acta Astronautica* 146 (2018), pp. 228–242.

[15] H. Gao. “Attitude Stabilization Control for Combined Spacecraft”. Available at https://kns.cnki.net/KCMS/detail/detail.aspx?dbname=CDFDLAST2021&filename=1020401662.nh. PhD thesis. Harbin Institute of Technology, 2019.

[16] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein. “Visualizing the loss landscape of neural nets”. In: *Advances in neural information processing systems* 31 (2018).

[17] J. Si, A. G. Barto, W. B. Powell, and D. Wunsch. *Handbook of learning and approximate dynamic programming*. Vol. 2. John Wiley & Sons, 2004.

[18] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. “Deep reinforcement learning that matters”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1. 2018.

[19] S. Fujimoto, H. Hoof, and D. Meger. “Addressing function approximation error in actor-critic methods”. In: *International conference on machine learning*. PMLR. 2018, pp. 1587–1596.

[20] P. Huang, Y. Lu, M. Wang, Z. Meng, Y. Zhang, and F. Zhang. “Postcapture attitude takeover control of a partially failed spacecraft with parametric uncertainties”. In: *IEEE Transactions on Automation Science and Engineering* 16.2 (2018), pp. 919–930.

5

CONCLUSIONS

This chapter offers a comprehensive overview of the content summary, answers to research questions and contributions of the PhD research. It further includes a reflective discussion on potential directions for future work, emphasizing opportunities for deeper investigation and continued development arising from the outcomes of this study.

5.1. Summary

This dissertation focuses on post-capture control with robotic arms in the context of active debris removal. It employs reinforcement learning to realize attitude control of the combined spacecraft after capture, and conducts an in-depth study on interpreting the performance and control results of reinforcement learning algorithms.

Chapter 1 introduces the background of Active Debris Removal, the motivation for using reinforcement learning for control and for interpreting RL algorithms, and a literature review covering ADR missions, conventional control, reinforcement-learning control, and interpretation methods. Research gaps are identified and the research questions are stated.

Chapter 2 establishes the critic match loss landscape visualization method. Using ADHDP as a sample algorithm, the method explains why a single RL algorithm can exhibit different performance on two systems.

Chapter 3 adapts the method to an off-policy, convergent setting using the Soft Actor–Critic (SAC) algorithm for the same attitude-control task. By comparing SAC's convergent result with ADHDP's divergent result through their corresponding loss landscapes, the method proves its usage across different RL paradigms.

Chapter 4 extends the method into a broader visualization framework that provides additional information for interpretation. Applied to several ADHDP variants, the framework explains the performance differences between variants.

5.2. Answers to the research questions

The research questions proposed in Chapter 1 can be answered.

1. How can the performance of RL control algorithms be interpreted in systems with uncertainty?

Reinforcement learning (RL) algorithms can be categorized based on how training data are used during learning. In online RL, the policy is updated directly from data generated through interaction with the environment during training. In contrast, some algorithms employ off-policy learning mechanisms that reuse previously collected data, for example through replay buffers.

Since online RL involves a more direct coupling between system interaction and policy updates, this study begins with the interpretation of online RL algorithms and subsequently extends the analysis to off-policy RL methods. Following this logic, this research question is addressed through two sub-questions:

a. How can an interpretation method be established to analyze the performance of online RL algorithms? (Chapter 2)

For actor–critic reinforcement learning algorithms, the critic neural network represents the value-function approximation that guides policy improvement. The behavior of the critic therefore reflects the underlying optimization process of the learning algorithm. Analyzing the critic therefore provides insight into the learning mechanism and the source of performance differences.

This work proposes a critic match loss landscape visualization method to interpret online RL. The method constructs a geometric representation of the critic optimization process by analyzing the evolution of critic parameters during training. Specifically, the critic parameters are recorded throughout the learning process and projected onto a two-dimensional plane obtained through principal component analysis (PCA), enabling visualization of the high-dimensional parameter space.

A fixed-target critic match loss is then defined using the system states from the final training episode and the corresponding temporal-difference targets. By combining different critic parameter values projected onto the PCA weight plane, the critic match loss for each weight combination is evaluated. The resulting 3-D loss landscape reveals the local geometry of the critic under the selected policy, while the 2-D optimization path qualitatively illustrates the evolution of training.

To enhance the reliability of the interpretation, several complementary analyses are introduced. Random-direction dimensionality reduction is further employed to examine the consistency of the interpretation under different projection directions. Temporal landscape snapshots are further constructed to illustrate how the loss landscape evolves during training. In addition, quantitative analysis of critic trajectory characteristics is performed to support the qualitative visualization results.

Using Action-Dependent Heuristic Dynamic Programming (ADHDP) as a representative online RL algorithm, the proposed framework reveals characteristic patterns in the critic update process. In particular, it explains why the same RL algorithm may exhibit significantly different control performance across systems with different dynamics. Rather than serving as a predictor of closed-loop stability, the critic match loss landscape provides a geometric interpretation of critic optimization dynamics, offering insight into training behavior and the sources of algorithm performance differences in online RL control under system uncertainty.

b. How can the interpretation method be applied to off-policy RL algorithms? (Chapter 3)

To extend the established visualization method to off-policy RL, adaptations are required because off-policy methods differ from online RL in data flow, update logic, and training structure. The Soft Actor–Critic (SAC) algorithm is selected as a representative case. Similar to RQ1.a, critic parameters are recorded during training and projected onto a two-dimensional plane using principal component analysis (PCA), enabling visualization of the high-dimensional parameter space.

For off-policy RL, the visualization framework is adapted by fixing both the training batch and the critic targets based on the final policy. This design is consistent with SAC's batch-based learning mechanism and entropy-regularized value formulation with twin critics. With the targets fixed, the critic match loss can be evaluated over different parameter combinations on the PCA plane, generating a three-dimensional loss landscape together with the projected trajectory of critic parameters throughout training.

The resulting visualization enables the analysis of critic optimization behavior in SAC training. Extensive simulations reveal that the static loss landscape alone

cannot serve as a solely reliable indicator of closed-loop control performance. Instead, meaningful interpretation arises from the combined analysis of loss geometry and the overleaid projected optimization trajectory. Different instability patterns can be identified through this joint representation. For example, some divergent cases exhibit strong directional domination in the PCA space, where critic updates evolve primarily along a single principal direction, producing ridge-like landscape structures. In other cases, the loss surface remains smooth, yet the optimization trajectory displays discontinuous relocations across separated regions of the parameter plane, indicating abrupt transitions in critic updates.

These observations demonstrate that the proposed visualization framework can diagnose distinct optimization patterns in off-policy RL training. By integrating the three-dimensional loss geometry with the two-dimensional parameter trajectory, the method provides a geometric interpretation of how critic parameters evolve during learning and offers a unified perspective for analyzing critic optimization dynamics across different reinforcement learning paradigms

2.How can the method identified in RQ1 be extended into a practical framework and applied to spacecraft attitude control?

This RQ is addressed through two sub-RQs.

a. Building on RQ1, how can the interpretation method be extended into a cohesive visualization framework?

The critic match loss landscape method established in RQ1 interprets the behavior of the critic in actor–critic reinforcement learning algorithms. However, the performance of RL algorithms is influenced not only by the critic but also by coupled components such as the actor and the environment dynamics. To provide a more complete view, this method is extended into a visualization framework that integrates multiple plots to interpret how these components interact and affect control performance. The framework contains four plots:

a) Critic match loss landscape. This plot helps explain how the critic network learns to approximate the cost function during training. It shows whether the critic reaches a stable fitting region or remains trapped in steep or flat areas of the loss surface.

b) Actor loss landscape. This plot explains how the critic network fits the TD targets during training. It shows whether the critic optimization reaches a stable region of the loss surface and reveals the geometric structure of the critic loss landscape.

c) Time–TD–actor weight trajectory. This plot illustrates how the actor weights change with time and TD error. It provides insight into the relationship between policy updates and the temporal behavior of the learning signal.

d) State–TD plot. This plot identifies which states contribute most to large TD errors. It helps locate unstable or poorly learned regions in the state space that influence overall control performance.

After answering the first sub-question, the second sub-question clarifies how the framework accounts for the performance of reinforcement learning algorithms.

b. How can this framework explain performance differences among ADHDP algorithm variants?

Using ADHDP for spacecraft attitude control as an example, the framework is applied to diagnose and compare the training behavior and failure mechanisms of four ADHDP variants: a basic version; a version with a target network; an advanced version with two stabilizers, Huber loss and cost scaling, together with target policy smoothing (TPS); and a version with the two stabilizers but without TPS. The visualization results show that the proposed framework can clearly reveal how each reinforcement learning technique affects the training process and control behavior. The critic and actor loss landscapes display how the target network, Huber loss, cost scaling, and TPS modify the shape and smoothness of the optimization surface. The TD- and state-based plots show how these changes influence the temporal evolution of learning signals and the stability of the controlled system. For example, the target network makes the TD evolution smoother but does not change the overall geometry of the loss surface. Huber loss and cost scaling reduce large TD peaks and produce a more rounded critic surface, while TPS further stabilizes the actor updates and suppresses action overestimation. These results show that the framework can visually connect RL techniques with effects on learning and control performance, making it a useful tool for understanding reinforcement learning methods and informing analysis.

5.3. Contributions

Contribution 1 — A method to characterize learning dynamics of online RL in dynamics and control

Reinforcement learning can fail to generalize across systems or under parameter changes. In actor–critic methods, the critic's approximation shapes the actor's updates and exploration, so understanding the critic is important for interpreting performance. This thesis proposes a critic match loss landscape visualization method to analyze the optimization dynamics of the critic during training. The method records critic parameters throughout training, projects them onto a low-dimensional parameter plane using principal component analysis, and evaluates a fixed-target critic loss constructed from reference states associated with selected policies. The resulting three dimensional surface with an overlaid two dimensional weight path characterizes the progression of training and the local optimization geometry, including flat regions and sharp valleys. These insights explain why the same RL algorithm may converge in some systems while exhibiting unstable learning dynamics in others, providing a geometric interpretation of critic optimization processes in online actor–critic reinforcement learning.

Contribution 2 — Extension of the Visualization Method to Off-policy Reinforcement Learning

Based on the established critic match loss landscape method, this study extends the visualization technique from online to off-policy reinforcement learn-

ing. The Soft Actor–Critic (SAC) algorithm is selected as a representative off-policy RL case to demonstrate the method's adaptability to batch-based and entropy-regularized learning structures. To accommodate the off-policy training paradigm, the visualization framework is adapted by fixing both the target computation and the training batch associated with the selected policy. This enables the reconstruction of the critic match loss over a principal component plane, producing a three-dimensional loss landscape together with the projected trajectory of critic parameters.Beyond methodological adaptation, the extended framework reveals characteristic patterns in critic optimization behavior during SAC training. Through extensive simulations, the visualization shows that static loss surfaces alone cannot determine control success. Instead, meaningful interpretation emerges from the combined analysis of loss geometry and overlaid projected parameter trajectories. Different instability mechanisms can be identified, including directional domination of critic updates and discontinuous parameter transitions across separated regions of the parameter space. These results demonstrate that the proposed visualization framework can interpret critic optimization dynamics in both online and off-policy reinforcement learning, providing a unified geometric perspective for analyzing learning behavior across different RL training paradigms.

Contribution 3 — A systematic Framework to Reveal the Training Evolution of RL

This thesis develops a visualization framework to study the training process of actor–critic reinforcement learning algorithms. The framework includes four plots: a 3-D critic match loss landscape, an actor loss surface with a frozen critic, a 3-D time–TD–actor weight trajectory, and a state–TD plot. Together, these plots show how the critic, actor, and TD error evolve during training and allow systematic comparison between algorithm variants. The framework provides a clear way to understand training behavior and to evaluate learning techniques beyond the use of reward curves.

5.4. Outlook

The interpretation framework based on training visualization can help the understanding of RL algorithms for control and inspire the design of the algorithms. There are significant opportunities for further investigation, including but not limited to:

Dynamic Visualization of the Loss Landscape for RL

Reinforcement learning interacts with the environment and updates online. Data and targets change during training, and the networks update accordingly. This is different from supervised learning, where data and labels are fixed. The method used here is adapted from supervised learning: data, targets, and weights are

collected at selected training points and projected into one loss landscape to indicate the trend. This projection is informative, but it does not capture the exact, instantaneous loss because the data distribution and weights are changing.

Future work can focus on dynamic visualization of the loss landscape. For example, synchronized data–target–weight snapshots are recorded at selected training times and provide a time-indexed sequence of landscapes with the corresponding optimization paths. This shows both the training trend and the actual loss associated with the data that generated each update, providing a closer view of how learning evolves evolves as training progresses.

Interpretation from the Stability Angle

Reinforcement learning for control can be viewed from two angles: the learning algorithm from the computer science and the closed-loop behavior from dynamics and control. This research adopts a landscape-based interpretation from the learning side, and also shows system changes through the state–TD plot.

Future work can integrate stability considerations more directly with interpretation. For example, stability rules or constraints can be included in the controller design.Their influence can then be observed in the visualization results. Vice versa, visualization can help explain divergence and guide the design of stability rules or training stabilizers. This design loop, design with stability, visualize the effect, and revise, can support more user-oriented and safety-guaranteed controller design.

Hardware In The Loop Demonstrations

Reinforcement learning controllers should be validated in physical experiments. For post-capture control of the combined spacecraft, a free-floating platform is needed. The interpretation framework can be used as a means for performance analysis in these experiments.

When noise, disturbances, and unmodeled uncertainties are present, the visualization trajectories may differ from simulation. Additional visualization tools may be developed to handle real data issues, such as time delay, output data constraint, making the interpretation of physical performance more reliable. Such tools can support the design and tuning of RL algorithms for dynamics and control in practical settings.

Acknowledgements

I would like to express my deepest gratitude to my two supervisors for giving me the opportunity to pursue a PhD and for all their guidance throughout this journey. I am deeply grateful to Prof. Eberhard Gill for his structural way of thinking and his approach to research management, from which I benefited enormously. His clarity in organizing complex research tasks has shaped my way of approaching scientific problems. I am sincerely thankful to Dr. Jian Guo for leading me into the promising and fascinating field of reinforcement learning and for helping me organize my research ideas and form a coherent research direction. His guidance has shaped the way I think about research and continuously motivated me to grow as an independent researcher.

I would also like to sincerely thank the members of my doctoral committee and all the reviewers who evaluated my work during the publication process. Your thoughtful and professional comments, as well as the constructive exchanges, helped me deepen my understanding of my research and refine it into more solid and meaningful contributions. I will carry this spirit of professionalism and rigor with me as I continue my future research journey.

I would like to thank my Master's supervisors, Prof. Jianhua Zheng and Prof. Mingtao Li, for guiding me into academic research and planting the seed of pursuing a doctoral degree. During my doctoral journey, there were moments of excitement and joy, moments of exploration and struggle, moments of heaviness and frustration, and moments of self-doubt. Whenever I felt that way, I would reread the acknowledgements from my Master's thesis and revisit our conversations on messaging apps. I am deeply grateful that you allowed me to have a period of life worth looking back on. This experience has become an internal source of strength, supporting me to persevere and complete my doctoral research.

This journey of studying abroad has brought me not only knowledge, but also invaluable friendships and life experiences. I am truly grateful to the colleagues and friends I have met here. Whether it was our discussions on academic research, the sharing of personal stories, or the enjoyment of delicious foods with a touch of local culture, your presence and companionship have broadened the horizons of my life. You have helped me grow into someone more open-minded and inclusive, able to look at this wonderful world with curiosity and appreciation.

I would also like to express my heartfelt gratitude to my father, Dr. Qingxin Liu, and my mother, Ms. Yuhui Zhang. From you, I have learned optimism and a genuine love for life, a sense of responsibility and dedication to one's work, care and commitment to family, and the drive to keep thinking, learning, and growing. Your values created a healthy environment for me to grow up in, shaping

a well-grounded character and giving me the courage and perseverance to face challenges in both life and research. I feel truly fortunate to have you as my parents. I love you very much.

I am also grateful to my parents-in-law. During our years of studying and living abroad, your encouragement, understanding, and thoughtfulness allowed us to settle into a foreign environment with peace of mind and live each day with steadiness.

I would like to thank my husband, Dr. Kai Wu. From Harbin, to Beijing, and then to Delft, we have known each other for more than ten years and shared over a decade together. During this time, I have come to understand you more deeply and love you more with each passing day. We are two people with different personalities, yet we share remarkably aligned values. Our differences allow us to reflect on each other and grow together, while our shared values give us mutual encouragement and the strength to strive forward, making us lifelong partners and comrades. Your presence has given me the courage to face the challenges of my PhD journey and life in general, and at the same time, provided a warm haven in moments of fatigue. I believe we will have a wonderful future together.

Finally, I would like to thank myself. I am grateful to the hopeful version of me, the optimistic version of me, and the version of me that never gave up. This has been not only a journey of research, but also a journey of self-discovery. I believe that although the scars left from pushing through the thorns may sometimes ache, they are honorable badges of my growth. This experience is a precious treasure in my life, one that will make me stronger, braver, and unstoppable—both in academia and in life.

Curriculum Vitæ

Jingyi Liu

20-01-1994 Born in Daqing, Heilongjiang, China.

Education

2013–2017 Bachelor of Science in Engineering
Harbin Institute of Technology, Harbin, China
Thesis: Approximate Modeling and Optimization of Orbital Dynamics Under Constant-Tangent Thrust
Supervisor: Prof. Gang Zhang

2017–2020 Master of Science in Engineering
University of Chinese Academy of Sciences, Beijing, China
Thesis: Transfer Trajectory Design and Optimization of Near-Earth Asteroid Sample Return Mission
Supervisors: Prof. Jianhua Zheng & Prof. Mingtao Li

2020–2025 PhD in Space Engineering
Delft University of Technology, the Netherlands
Thesis: Interpreting Reinforcement Learning for Post-Capture Control in Space Debris Removal
Promotors: Prof. Eberhard Gill & Dr. Jian Guo

List of Publications

Publications Related to This Dissertation

4. Jingyi Liu, Jian Guo, and Eberhard Gill, “A Loss Landscape Visualization Framework for Interpreting Reinforcement Learning: An ADHDP Case Study,” submitted to *Acta Astronautica* and available as a preprint on arXiv. DOI: https://doi.org/10.48550/arXiv.2603.14600

3. Jingyi Liu, Jian Guo, and Eberhard Gill, “Adapting Critic Match Loss Landscape Visualization to Off-policy Reinforcement Learning,” revision submitted to *Astrodynamics* and available as a preprint on arXiv. DOI: https://doi.org/10.48550/arXiv.2603.14589

2. Jingyi Liu, Jian Guo, and Eberhard Gill, “Visualizing Critic Match Loss Landscapes for Interpretation of Online Reinforcement Learning Control Algorithms,” revision submitted to *Acta Astronautica* and available as a preprint on arXiv. DOI: https://doi.org/10.48550/arXiv.2603.14535

1. Jingyi Liu, Jian Guo and Eberhard Gill, “Online policy iteration ADP-based control of post-capture combined spacecraft without inertia identifications,” 73rd International Astronautical Congress, Paris, 18–22 September 2022.

Publications Not Related to This Dissertation

2. Jing Quan, Liu Jingyi, Li Mingtao. "Hybrid optimization method for lunar gravity assist transfer trajectory to near-Earth asteroids". Journal of Harbin Institute of Technology, 2022, 54(11): 31-37. DOI: https://doi.org/10.11918/202109123.

1. Liu Jingyi, Zheng Jianhua and Li Mingtao. "Dry Mass Optimization for Impulsive Transfer Trajectory of Near Earth Asteroid Sample Return Mission". Astrophysics and Space Science, 2019, 364:215. DOI: https://doi.org/10.1007/s10509-019-3703-0

ISBN:978-94-6518-282-7