POLITECNICO DI TORINO

MASTER's Degree in SPACE ENGINEERING

MASTER's Degree Thesis

Supervised Learning Algorithm for Autonomous Rendezvous in Space

Supervisors

Prof. ELISA CAPELLO

Ing. SIMONE CHESI

Candidate

ALBERTO PIREDDA

2025/2026

Abstract

The rapid growth of the satellites in Earth orbit and the prospect of increasingly dense constellations, is reshaping the operational landscape of spaceflight. In this environment, ground-intensive approaches to fleet operations, inspection, and anomaly response become progressively less scalable. This trend motivates autonomous rendezvous and proximity operations as enabling technologies for routine inspection and servicing, as well as for safer and more efficient operations in space missions. At the same time, it imposes stricter requirements on onboard guidance: algorithms must be robust, constraint-aware, and computationally efficient to support repeatable operations with limited human intervention.

Autonomy, however, cannot be assessed solely through nominal simulation performance. A flight-relevant guidance solution is expected to provide consistent convergence, enforce feasibility under limited control authority, tolerate uncertainties in dynamics and vehicle properties, and run with bounded latency on onboard computers. These requirements are particularly demanding in the far-range phase when low-thrust propulsion is adopted: maneuvers extend over long horizons, control authority is tightly bounded, and small modeling errors can accumulate into significant trajectory deviations. As a result, the design of guidance policies for far-range low-thrust rendezvous becomes a system-level problem at the intersection of astrodynamics, guidance design, verification, and onboard deployability.

This thesis proposes a structured workflow to design and evaluate autonomous far-range rendezvous guidance under low-thrust actuation. The approach starts from a formal problem definition including objectives, constraints, and a hierarchy of dynamical models used for nominal and perturbed testing. A reference guidance policy is implemented and exercised in closed loop across diverse scenarios to produce a traceable set of state–action trajectories. The resulting dataset is curated through physically meaningful feature construction and consistent normalization, complemented by filtering and balancing procedures to preserve representative coverage of both thrusting and coasting conditions.
Using these expert demonstrations, a compact surrogate policy is learned to approximate the baseline mapping from relative state information to commanded acceleration in a computationally efficient form suitable for real-time integration. The surrogate is designed to interface cleanly with simulation and software workflows, enabling deterministic inference and straightforward substitution within a closed-loop guidance architecture. The evaluation plan relies on mission-oriented metrics at both command level and trajectory level, with emphasis on convergence

reliability, time-to-handover behavior, control-effort proxies, and constraint compliance under nominal conditions and under perturbations representative of realistic operations.

"Somewhere, something incredible is waiting to be known."

— Carl Sagan

Acknowledgements

I would like to express my sincere gratitude to Professor Elisa Capello for her guidance, availability, and constant support throughout the development of this thesis. Her scientific rigor, valuable advice, and thoughtful supervision were essential in helping me approach each stage of this work with method and awareness.
I would also like to warmly thank Engineer Simone Chesi for his continuous support, technical insight, and to give me this opportunity.
I would like to thank all the members of CUS GNC, with whom I had the pleasure of sharing ideas, experiences, and moments of both technical and personal growth.
I would also like to express my deepest gratitude to my family for their constant love, encouragement, and unwavering support throughout my academic journey. Their trust in me, together with their patience and closeness, has always given me the strength and motivation to face every challenge along the way.
Finally, I would like to thank my friends for their presence, support, and the many moments shared during these years. Their encouragement, understanding, and companionship have made this journey lighter and more meaningful, both academically and personally.

Table of Contents

List of Tables

List of Figures

Acronyms

COE
classical orbital elements

DCM
direction cosine matrix

ECEF
Earth-Centered Earth-Fixed

ECI
Earth-Centered Inertial

EOP
Earth orientation parameters

ICRF
International Celestial Reference Frame

LVLH
local-vertical local-horizontal

MEE
modified equinoctial elements

MLP
multilayer perceptron

RAAN
right ascension of the ascending node

RTN

radial-tangential-normal

TAI

International Atomic Time

TT

Terrestrial Time

UT1

Universal Time 1

UTC

Coordinated Universal Time

DAgger

Dataset Aggregation

Chapter 1

Introduction

Orbital rendezvous is a fundamental capability for a wide range of space missions, including servicing, inspection, debris interaction, in-orbit assembly, logistics, and mission extension. In many of these applications, the guidance system must steer an active spacecraft, hereafter referred to as the *chaser*, toward a target object while satisfying operational constraints, respecting limited actuation authority, and preserving a terminal state suitable for handover to a subsequent proximity-operations layer. When such maneuvers are performed under low-thrust propulsion, the problem becomes particularly challenging because the available acceleration is small, the maneuver duration is long, and even modest guidance errors can accumulate over many orbital revolutions.
This thesis focuses on *far-range low-thrust rendezvous*, i.e. the guidance problem arising before terminal proximity operations, when the chaser is still separated from the target by large relative distances and must progressively align orbit geometry, relative phase, and terminal approach conditions over long horizons. In this regime, the guidance law must remain effective over many updates, must cope with the multiscale coupling between orbital geometry and along-track phasing, and must remain numerically robust under bounded thrust authority. These characteristics make the problem particularly suitable for studying the interaction between model-based astrodynamics and data-driven policy approximation.

The work presented in this thesis is motivated by the following observation. On the one hand, model-based guidance laws provide physical interpretability, explicit constraint handling, and predictable failure analysis. On the other hand, their direct online evaluation may become cumbersome when the guidance logic includes orbital-element conversion, sensitivity-based control directions, mode sequencing, and supervisory safeguards. A natural research direction is therefore to ask whether a compact neural surrogate can learn the *command-selection map* of a reliable model-based expert, while preserving enough consistency and robustness to remain meaningful in closed-loop operation.

The central idea investigated here is precisely this: rather than replacing the entire guidance architecture with a black-box policy, the learning problem is restricted to the expert command generator itself. The surrounding simulation and control structure remains deterministic and model-based, while the neural network is tasked with approximating the mapping from relative state information to commanded acceleration. This framing turns the surrogate into a *drop-in policy module* embedded in a physically structured environment, rather than into a fully end-to-end autonomous controller.

1.1 Motivation and Research Context

Low-thrust propulsion has become increasingly relevant for modern space missions because it offers high propellant efficiency and enables long-duration transfer and rendezvous maneuvers that would be impractical under purely impulsive assumptions. However, the same feature that makes low-thrust attractive from a mission-design perspective also complicates guidance. Because the control authority is weak, the resulting trajectories are long, strongly path-dependent, and sensitive to persistent directional errors. In the far-range rendezvous regime, geometry correction and phase correction are not naturally synchronized, and the guidance law must often decide how to trade shape, plane, and timing alignment over extended horizons.

This difficulty makes low-thrust rendezvous a useful case study for surrogate-guidance research. Unlike short-horizon control tasks, it exposes the gap between open-loop imitation accuracy and closed-loop operational reliability. A surrogate that predicts expert commands reasonably well on demonstrated states may still fail when used online, because small action discrepancies alter the visited-state distribution and progressively move the policy away from the regime represented in the training data. This phenomenon is well known in imitation learning and sequential decision-making, where distribution shift and error compounding can invalidate otherwise promising regressors [1].

From this perspective, the present thesis is not only about whether a neural network can fit expert demonstrations, but about whether such a surrogate can remain useful inside a realistic closed-loop rendezvous architecture. The problem is therefore as much about *interface consistency*, *deployment semantics*, and *failure modes* as it is about regression accuracy.

1.2 State of the Art

1.2.1 Model-Based Low-Thrust Guidance

The classical solution space for low-thrust orbital transfer and rendezvous is dominated by model-based guidance and optimization methods. Direct and indirect optimal-control approaches can produce high-quality trajectories, but they may be computationally demanding and sensitive to initialization when used in long-horizon many-revolution settings. For this reason, a significant body of work has also focused on guidance laws that trade strict optimality for computational tractability and online interpretability.
Within this family, Q-law-type approaches are particularly relevant to the present work. In these methods, the control is chosen to decrease a Lyapunov-like or weighted error potential in orbital-element space, often through local control sensitivities and heuristics for sequencing or coasting [2, 3]. Such methods are attractive because they combine physical transparency with relatively low online computational burden. More recent studies have continued to use Q-law either directly or as a seed for more refined low-thrust trajectory design pipelines, confirming its practical value in many-revolution transfer problems [4, 5].

A second important ingredient in element-based low-thrust guidance is the choice of orbital representation. Modified Equinoctial Elements (MEE) are widely adopted because they provide a numerically robust alternative to classical orbital elements in near-circular and near-equatorial regimes, while preserving a convenient mapping between RTN acceleration and element-rate dynamics [6]. This representation is particularly well suited to guidance architectures that decompose the rendezvous objective into orbit size, shape, plane, and phase components.
In addition, the RTN frame remains a natural coordinate system for command interpretation in rendezvous guidance. Tangential acceleration primarily affects energy and semi-major-axis evolution, normal acceleration drives plane correction, and radial acceleration contributes to geometry shaping [7]. This directional decomposition is one of the reasons why model-based guidance laws often remain easier to diagnose and tune than purely inertial black-box policies.

1.2.2 Learning-Based Surrogates and Imitation Learning

In recent years, machine learning has increasingly been explored as a way to approximate expensive guidance, optimization, or control modules with compact inference-time models. In the context of sequential decision problems, however, learning from demonstrations introduces a known difficulty: policies trained only on expert-generated states may degrade significantly when deployed online, because

their own actions alter the state distribution. This mismatch between training and deployment distributions can lead to compounding error over time [1].

For orbital guidance, this issue is especially important in long-horizon low-thrust settings. A one-step prediction error that appears small in open-loop evaluation may still produce meaningful trajectory drift when propagated over many guidance updates. The challenge is therefore not merely to reduce regression loss, but to preserve the directional and magnitude semantics that matter for closed-loop convergence.
These considerations motivate a cautious learning architecture. Rather than attempting to learn the entire rendezvous problem end-to-end, the surrogate developed in this thesis is deliberately embedded in a deterministic model-based shell. This design preserves physically meaningful preprocessing, explicit thrust scaling, deterministic frame mappings, and traceable supervisory logic, while delegating only the command-selection map to the neural approximator. The result is a hybrid model-based / learned framework in which the neural network is treated as a surrogate module rather than as a replacement for the full guidance system.

1.3 Problem Framing and Thesis Objective

The main objective of this thesis is to investigate whether a neural surrogate can approximate a robust model-based expert guidance law for far-range low-thrust rendezvous in a form that remains computationally efficient, physically interpretable, and operationally meaningful in closed loop.

To address this objective, the work is organized around the following steps:

1. formalize the far-range rendezvous problem, including state definitions, performance metrics, perturbation models, and terminal criteria;

2. design and implement a model-based expert guidance law that is robust, bounded, and suitable both for closed-loop use and for demonstration generation;

3. construct a dataset pipeline whose feature and label semantics are fully consistent with the deployed surrogate interface;

4. train a compact neural policy to approximate the expert command generator;

5. evaluate the surrogate in both open-loop and closed-loop settings, with particular attention to distribution shift, error compounding, and stability under deployment.

The goal is therefore not to claim that the learned surrogate outperforms the model-based expert in every respect, nor to propose a fully autonomous learned rendezvous controller. Rather, the purpose of this thesis is to determine to what extent a compact learned surrogate can reproduce the expert guidance behavior with sufficient consistency to remain useful in closed-loop operation, and to identify the mechanisms that limit this transfer from open-loop imitation to trajectory-level autonomy.

1.4 Scope and Delimitations

The work deliberately focuses on the *far-range* portion of the rendezvous problem. Terminal proximity operations, docking logic, and terminal vision-based guidance are outside the scope of the present study. Similarly, the thesis does not pursue a globally optimal solution of the low-thrust rendezvous problem. The expert guidance law is designed as a robust and interpretable baseline suitable for simulation-driven learning, not as a provably optimal or formally globally stable hybrid supervisor.

On the learning side, the thesis does not attempt to learn the entire spacecraft dynamics or mission logic. The surrogate only approximates the expert command-selection map. Orbital propagation, frame construction, thrust-to-acceleration conversion, and terminal logic remain model-based and deterministic. This delimitation is intentional, because it permits a cleaner analysis of where learning helps, where it fails, and how deployment semantics affect closed-loop behavior.

Finally, while dataset aggregation and iterative retraining strategies are considered during development, the principal scientific narrative of the thesis is centered on the baseline expert-to-surrogate pipeline and on the observed discrepancy between open-loop agreement and closed-loop robustness. In this sense, the thesis is as much an analysis of the *limits* of straightforward surrogate guidance as it is a demonstration of its promise.

1.5 Main Contributions

The main contributions of the thesis can be summarized as follows:

1. a structured formulation of the far-range low-thrust rendezvous problem, including operational metrics and termination logic suitable for trajectory-level comparison;

2. a Q-law-inspired hybrid expert guidance law that combines element-based geometry closure, gated phase correction, bounded actuation, and persistence-based handover detection;

3. a deployable neural surrogate architecture defined not only by an MLP, but by a complete inference-time contract including preprocessing, clipping, confidence-aware attenuation, and deterministic frame remapping;
4. a traceable dataset-generation and curation pipeline based on chunked artifacts, frozen scalers, rollout-level semantics, and semantic consistency between training and deployment;
5. an evaluation framework that distinguishes open-loop command-level agreement from closed-loop trajectory-level performance, thereby exposing the role of distribution shift and error compounding in learned low-thrust guidance.

Taken together, these contributions support a central thesis claim: open-loop imitation quality is a necessary but insufficient condition for reliable closed-loop guidance in long-horizon low-thrust rendezvous. What matters equally is the consistency of the deployed policy contract and the extent to which surrogate errors remain bounded under the state distribution induced by the surrogate itself.

1.6 Thesis Structure

The remainder of the thesis is organized as follows.

Chapter 2 formalizes the far-range low-thrust rendezvous problem, introduces the dynamical and operational assumptions, and defines the metrics used throughout the work.

Chapter 3 presents the model-based expert guidance law. The expert is formulated as a Q-law-inspired hybrid strategy combining geometry closure, gated phase correction, feasibility enforcement, and persistence-based handover detection.

Chapter 4 defines the neural surrogate policy as a deployable module. It introduces the history-stacked RTN feature representation, the MLP architecture, and the deterministic inference-time wrapper used to convert the network output into a feasible physical acceleration command.

Chapter 5 describes the dataset-generation and curation pipeline, including feature construction, label normalization, rollout-level variation strategy, chunked artifact management, and deterministic validation checks.

Chapter 6 compares expert and surrogate guidance through both action-level and trajectory-level metrics, with emphasis on the gap between open-loop imitation and closed-loop behavior.

1.7 Chapter Summary

This chapter introduced the research context and motivation of the thesis, positioned the work with respect to model-based low-thrust guidance and learning-based

surrogate policies, and defined the scope and objective of the study. The central idea is to approximate a robust model-based expert guidance law with a compact neural surrogate embedded in a deterministic guidance architecture, and to evaluate this surrogate not only in terms of one-step imitation accuracy but also in terms of closed-loop trajectory behavior. The next chapter turns this high-level motivation into a formal problem statement by defining the far-range rendezvous regime, the relevant state representations, the considered dynamics, and the performance metrics used throughout the thesis.
A particular emphasis is placed on the distinction between command-level imitation quality and closed-loop guidance reliability, since this gap ultimately determines the practical value of the surrogate policy.

Chapter 2

Problem Definition and Formulation

This chapter formalizes the far-range low-thrust rendezvous problem studied in this thesis. The objective is to define, in a self-contained way, (i) the relative state and the operational definition of "far-range", (ii) the dynamical models and perturbations considered in simulation, (iii) operational constraints and safety requirements, and (iv) convergence and termination criteria. These definitions provide a consistent language for all subsequent chapters and ensure that comparisons between guidance implementations are carried out under identical assumptions.

2.1 Definition of Far-Range Rendezvous and Performance Metrics

Orbital rendezvous is the controlled process by which an active spacecraft (the *chaser*) is guided toward a second object (the *target*) in order to achieve a prescribed terminal relative state suitable for subsequent operations (e.g., inspection, docking, capture, servicing, or handover to a proximity guidance layer). In this thesis, the focus is on *far-range* rendezvous under *low-thrust* actuation, where the maneuver typically spans long time horizons and the available acceleration authority is bounded and relatively small.

2.1.1 Orbital Elements

In addition to Cartesian inertial states $(\mathbf{r}, \mathbf{v})$, the analysis and the definition of rendezvous errors benefit from orbital-element representations, which provide an intuitive decomposition of the motion into *orbit size*, *shape*, *plane*, and *phase*. Two

element sets are used throughout the thesis: (i) COE, mainly for interpretation and reporting, and (ii) MEE, adopted as a numerically robust representation to define geometry-related errors. The orbital-mechanics definitions and notation adopted in this chapter follow standard astrodynamics conventions.[8]

Classical Orbital Elements (COE)

The osculating Keplerian elements are

$$(a,\ e,\ i,\ \Omega,\ \omega,\ \nu), \tag{2.1}$$

where a is the semi-major axis (setting the orbit size), e is the eccentricity (quantifying the deviation from circularity and thus the orbit shape), i is the inclination, i.e., the angle between the orbital plane and the reference equatorial plane, Ω is the right ascension of the ascending node (RAAN), defined as the inertial angle from a fixed reference direction to the line of nodes measured in the reference plane, ω is the argument of periapsis, defined in the orbital plane as the angle from the ascending node to the periapsis direction, and ν is the true anomaly, defined as the instantaneous angle in the orbital plane from periapsis to the position vector.

Given the inertial state $(\mathbf{r}, \mathbf{v})$, the fundamental vectors are:

$$\mathbf{h} \triangleq \mathbf{r} \times \mathbf{v}, \qquad h \triangleq \|\mathbf{h}\|, \tag{2.2}$$

$$\mathbf{e} \triangleq \frac{\mathbf{v} \times \mathbf{h}}{\mu} - \frac{\mathbf{r}}{\|\mathbf{r}\|}, \qquad e \triangleq \|\mathbf{e}\|, \tag{2.3}$$

with μ the Earth's gravitational parameter. The specific orbital energy is

$$\varepsilon \triangleq \frac{\|\mathbf{v}\|^2}{2} - \frac{\mu}{\|\mathbf{r}\|}, \tag{2.4}$$

so that $a = -\mu/(2\varepsilon)$ for bounded orbits. Angles (i, Ω, ω, ν) are obtained by standard geometric constructions from $\mathbf{h}$, the line of nodes, and the eccentricity vector. In this thesis, COEs are computed from $(\mathbf{r}, \mathbf{v})$ using a numerically robust RV-to-COE conversion routine, consistently with standard astrodynamics definitions.[8]

Modified Equinoctial Elements (MEE)

To avoid the singularities of COEs at small eccentricity ($e \to 0$) and small inclination ($i \to 0$), we also use Modified Equinoctial Elements, following the classical equinoctial-element formulation and its modified nonsingular variant.[9, 6]

$$\mathbf{x}_{\text{mee}} \triangleq \begin{bmatrix} p & f & g & h & k & L \end{bmatrix}^\top, \tag{2.5}$$

where

$$p \triangleq a(1-e^2), \tag{2.6}$$

$$f \triangleq e\cos(\Omega+\omega), \qquad g \triangleq e\sin(\Omega+\omega), \tag{2.7}$$

$$h \triangleq \tan\left(\frac{i}{2}\right)\cos\Omega, \qquad k \triangleq \tan\left(\frac{i}{2}\right)\sin\Omega. \tag{2.8}$$

The angular variable L is a longitude-like parameter. For geometry tracking and reporting, a common choice is the *true longitude*

$$L \triangleq \Omega+\omega+\nu. \tag{2.9}$$

The adopted (p, f, g, h, k, L) parameterization follows the standard MEE definition widely used for nonsingular orbital representation and perturbation analysis.[6]

Osculating versus mean-like quantities. Unless otherwise stated, orbital elements in this thesis are *osculating* elements computed from the instantaneous Cartesian state $(\mathbf{r}, \mathbf{v})$. In perturbed motion, osculating elements exhibit short-period variations. For this reason, when phasing variables are used as long-horizon indicators, "mean-like" quantities (e.g., mean longitude λ or filtered surrogates) are preferred to reduce sensitivity to short-period oscillations while preserving the long-term trend relevant to far-range rendezvous, consistent with standard orbital-element practice.[8]

Mean Longitude for Phasing (Definition)

Far-range rendezvous requires closing both *geometry* (orbit size/shape/plane) and *phasing*. A convenient phasing indicator is the *mean longitude*:

$$\lambda \triangleq \Omega+\omega+M, \tag{2.10}$$

where M is the mean anomaly. Given (ν, e), the eccentric anomaly E and mean anomaly M satisfy:

$$E = 2\arctan\left(\sqrt{\frac{1-e}{1+e}}\,\tan\frac{\nu}{2}\right), \qquad M = E - e\sin E. \tag{2.11}$$

The variable λ is "mean-like" and is less affected by short-period oscillations than the true longitude, which makes it well suited to define a phasing error over long horizons in far-range rendezvous analysis.[8]

2.1.2 Relative State and Element-Based Errors

This section defines the relative quantities used to assess rendezvous progress and formalizes orbit-geometry errors in terms of Modified Equinoctial Elements (MEE), which provide a robust, near-singularity-free description of orbital differences.

Inertial Relative State

Let $(\mathbf{r}_c^I, \mathbf{v}_c^I)$ and $(\mathbf{r}_t^I, \mathbf{v}_t^I)$ denote the inertial states of chaser and target. The inertial relative position and velocity are defined as

$$\boldsymbol{\rho}^I(t) \triangleq \mathbf{r}_c^I(t) - \mathbf{r}_t^I(t), \qquad \dot{\boldsymbol{\rho}}^I(t) \triangleq \mathbf{v}_c^I(t) - \mathbf{v}_t^I(t). \tag{2.12}$$

Two fundamental scalar indicators are the range and the relative speed,

$$\rho(t) \triangleq \|\boldsymbol{\rho}^I(t)\|, \qquad v_{\mathrm{rel}}(t) \triangleq \|\dot{\boldsymbol{\rho}}^I(t)\|, \tag{2.13}$$

together with the line-of-sight (radial) closing rate

$$\dot{\rho}(t) \triangleq \frac{\boldsymbol{\rho}^I(t)^\top \dot{\boldsymbol{\rho}}^I(t)}{\|\boldsymbol{\rho}^I(t)\|}. \tag{2.14}$$

These quantities are used to define terminal handover conditions and to monitor safety-related behavior (e.g., avoidance of excessive closing speeds).

Notation consistency. In the remainder of this chapter, the superscript $(\cdot)^I$ is omitted when no ambiguity arises; all relative quantities are assumed expressed in the inertial frame unless explicitly stated.

Motivation for Element-Based Errors

While Cartesian relative variables are essential for terminal verification, far-range rendezvous is strongly driven by *orbital geometry*: size, shape, and plane alignment, as well as phasing along the orbit. Element-based errors provide an interpretable decomposition of these aspects and support long-horizon monitoring where $\rho(t)$ may not decrease monotonically due to orbital mechanics and perturbations.

In this thesis, element-based errors are defined using Modified Equinoctial Elements, which are well suited for near-circular and/or near-equatorial orbits where classical Keplerian elements may become ill-conditioned.[9, 6]

Interpretation. The element p captures orbit size (through semi-latus rectum), (f, g) encode the eccentricity vector (shape and periapsis orientation), and (h, k) encode the orbital plane (inclination and node) in a nonsingular way. This makes MEE particularly convenient for defining smooth geometry errors and for separating orbit-size, shape, and plane effects in a numerically robust form.[6]

Angle Wrapping Convention

For any angular variable α, differences are computed using the wrapped value in $(-\pi, \pi]$:

$$\mathrm{wrap}_\pi(\Delta\alpha) \triangleq \Delta\alpha - 2\pi \left\lfloor \frac{\Delta\alpha + \pi}{2\pi} \right\rfloor . \tag{2.15}$$

This avoids discontinuities at 2π and ensures that the smallest signed angular error is used.

MEE Error Vector and Geometry Error Metric

Let $\mathbf{x}_{\mathrm{mee},c}$ and $\mathbf{x}_{\mathrm{mee},t}$ denote the MEE vectors of chaser and target computed from the current inertial states. The MEE error is defined as:

$$\Delta\mathbf{x}_{\mathrm{mee}} \triangleq \mathbf{x}_{\mathrm{mee},c} - \mathbf{x}_{\mathrm{mee},t} = \begin{bmatrix} \Delta p & \Delta f & \Delta g & \Delta h & \Delta k & \Delta L \end{bmatrix}^\top , \tag{2.16}$$

where the longitude difference is evaluated using angle wrapping:

$$\Delta L \triangleq \mathrm{wrap}_\pi(L_c - L_t) . \tag{2.17}$$

For far-range geometry tracking, the primary geometry error vector is

$$\Delta\mathbf{y} \triangleq \begin{bmatrix} \Delta p & \Delta f & \Delta g & \Delta h & \Delta k \end{bmatrix}^\top , \tag{2.18}$$

excluding the phase component. A compact scalar geometry error is defined by scaling p with a reference value p_{ref} and leaving the remaining equinoctial components unscaled:

$$e_{\mathrm{geom}} \triangleq \sqrt{\left(\frac{\Delta p}{p_{\mathrm{ref}}}\right)^2 + (\Delta f)^2 + (\Delta g)^2 + (\Delta h)^2 + (\Delta k)^2}, \qquad p_{\mathrm{ref}} \triangleq \max(p_t, \epsilon_p), \tag{2.19}$$

where ϵ_p is a small positive constant used to avoid division by zero. This definition provides a dimensionless indicator of orbit similarity in terms of size/shape/plane, suitable for monitoring convergence over long horizons.

Phasing Error (Element-Based)

A separate quantity is required to capture *phasing* along the orbit. In an element-based view, phasing can be represented by a longitude-like angular error, computed with wrapping. In this thesis, the phase mismatch is expressed as an angular difference (e.g., a mean-like longitude or a filtered longitude-like variable derived from osculating elements), denoted here generically as $\Delta\lambda$:

$$\Delta\lambda \triangleq \mathrm{wrap}_\pi(\lambda_c - \lambda_t) . \tag{2.20}$$

The specific definition of λ used in the implementation is chosen to be robust to short-period oscillations, so that $\Delta\lambda$ captures long-term phasing mismatch relevant to far-range closure.

Scaling and interpretability. The geometry metric in (2.19) combines heterogeneous components. The normalization of Δp by p_{ref} yields a dimensionless contribution comparable in magnitude to (f, g, h, k), which are already dimensionless. Alternative scalings (e.g., weighting matrices) can be adopted to reflect mission priorities (plane closure vs shape closure). In this thesis, the unweighted form is preferred to avoid embedding task-specific heuristics in the metric definition and to keep comparisons between guidance policies transparent.

Near-circular and near-equatorial regimes. A key motivation for using MEE is numerical robustness when $e \to 0$ and/or $i \to 0$. In such regimes, COE-based errors that involve ω or Ω may become ill-conditioned, whereas (f, g, h, k) remain well-behaved.[9, 6] For this reason, geometry closure is assessed primarily in MEE space. COEs are still reported for interpretability (e.g., Δa, Δi, $\Delta\Omega$), but they are not used as primary convergence criteria.

Relationship to far-range guidance structure. In far-range, it is often beneficial to treat geometry closure and phasing closure as partially decoupled: geometry closure aims at matching orbit size/shape/plane, while phasing closure aligns mean longitudes (or equivalent mean-like surrogates). This separation is aligned with Lyapunov-based far-range guidance strategies, where the Lyapunov function is often constructed from geometry and phase error components with different time scales.

Separation of roles. In summary, $(\rho, v_{\text{rel}}, \dot{\rho})$ are used for terminal verification and safety monitoring, while $(\Delta\mathbf{y}, \Delta\lambda)$ provide an interpretable decomposition of far-range progress in geometry and phase.

2.1.3 Relative State and Range-Based Definitions

For completeness and to support later diagnostics, we restate that range-based scalars are obtained directly from the inertial relative state. Specifically, the relative position and velocity in the inertial frame F_I are

$$\rho^I(t) \coloneqq r_c^I(t) - r_t^I(t), \qquad \dot{\rho}^I(t) \coloneqq v_c^I(t) - v_t^I(t), \tag{2.21}$$

and the scalar range is

$$\rho(t) \coloneqq \|\rho^I(t)\|. \tag{2.22}$$

The closing rate along the line of sight is given by (2.14). These quantities compress the relative motion into interpretable scalars: distance to target and instantaneous rate of approach/recession along the line of sight.

In far-range, $\rho(t)$ may vary non-monotonically due to orbital geometry and perturbations; therefore, range-based measures must be complemented by trajectory-level indicators (Section 2.1.5) and by element-based errors (Section 2.1.2).

Depending on mission conventions, the relative state may also be expressed in a target-centered rotating frame (e.g., LVLH/RTN) or via differences in orbital elements. The formulation in this chapter remains general and does not depend on a specific coordinate choice; the key requirement is that the chosen representation can be mapped into guidance commands and performance metrics consistently.

2.1.4 Definition of the Far-Range Regime and Handover Set

The rendezvous timeline is commonly partitioned into phases (far-range, mid-range, near-range) to reflect changes in dominant constraints and required modeling fidelity. In this work, far-range is characterized by:

- large initial separation and typically weak geometric constraints compared to near-range;
- long-duration maneuvers, often spanning multiple orbital revolutions;
- limited control authority, so that convergence must be achieved gradually and robustly;
- the primary objective of shaping the relative orbit/geometry for a safe and efficient transition to later phases.

Operational characterization (order-of-magnitude). While the formulation is kept general, the term *far-range* refers here to rendezvous scenarios in which the initial separation is typically much larger than the final handover tolerance, and the maneuver spans multiple orbital revolutions under low available acceleration. In this regime, $\rho(t)$ is not required to decrease monotonically, and progress is more meaningfully tracked through trajectory-level indicators (e.g., geometry/phasing proxies and feasibility metrics) rather than instantaneous range reduction.

Operationally, far-range rendezvous is defined as the interval

$$t \in [t_0, t_f], \tag{2.23}$$

where t_0 is the start time of the autonomous far-range guidance and t_f is the time at which the system reaches a prescribed handover set $\mathcal{X}_f$ in the relative-state space. A generic definition of $\mathcal{X}_f$ is:

$$\mathcal{X}_f \triangleq \{(\boldsymbol{\rho}, \dot{\boldsymbol{\rho}}) : \|\boldsymbol{\rho}\| \leq \rho_{\text{tol}},\ \|\dot{\boldsymbol{\rho}}\| \leq v_{\text{tol}},\ \phi(\boldsymbol{\rho}, \dot{\boldsymbol{\rho}}) \leq \phi_{\text{tol}}\}, \tag{2.24}$$

where ρ_{tol} and v_{tol} are mission-defined tolerances, and $\phi(\cdot)$ denotes an optional additional geometric or energetic constraint (e.g., approach corridor, relative-orbit alignment proxy, or bounded relative energy) required to ensure compatibility with the next-phase guidance logic. Typical examples of $\phi(\cdot)$ used to enforce handover compatibility include:

- **Approach corridor / cone:** a bound on the relative position direction in a target-centered frame, e.g., $\phi(\rho, \dot{\rho}) = \angle(\rho, \hat{c})$ with $\angle(\rho, \hat{c}) \leq \phi_{\text{tol}}$ for a prescribed corridor axis $\hat{c}$;
- **Relative-orbit alignment proxy:** a constraint on geometry/phasing residuals, e.g., $\phi(\rho, \dot{\rho}) = e_{\text{geom}}$ or a combined metric $(e_{\text{geom}}, |\Delta\lambda|)$;
- **Bounded relative energy / closing behavior:** a constraint limiting excessive closing speed near handover, e.g., $\phi(\rho, \dot{\rho}) = \|\dot{\rho}\|$ or an energy-like proxy to prevent fragile terminal crossings.

The rendezvous success event for the far-range phase is then defined as:

$$(\boldsymbol{\rho}(t_f), \dot{\boldsymbol{\rho}}(t_f)) \in \mathcal{X}_f \quad \text{for some } t_f \leq t_{\max}, \tag{2.25}$$

where $t_{\max}$ is a maximum allowable duration reflecting operational constraints (timeline, power/thermal constraints, communication windows, or mission scheduling).

2.1.5 Performance Metrics: Mission-Oriented Evaluation

Because far-range low-thrust maneuvers are long and guidance acts continuously, evaluation must extend beyond instantaneous command matching and focus on trajectory-level outcomes and feasibility. This thesis adopts the following families of performance metrics.

(i) Convergence and success rate. The most fundamental metric is whether the handover set is reached within the allowable time:

$$I_{\text{succ}} \triangleq \begin{cases} 1, & \exists\, t_f \leq t_{\max} \text{ such that } (\boldsymbol{\rho}(t_f), \dot{\boldsymbol{\rho}}(t_f)) \in \mathcal{X}_f, \\ 0, & \text{otherwise.} \end{cases} \tag{2.26}$$

Over an ensemble of scenarios (e.g., Monte Carlo campaign), a key robustness indicator is the empirical success probability:

$$\hat{p}_{\text{succ}} \triangleq \frac{1}{N} \sum_{i=1}^{N} I_{\text{succ}}^{(i)}. \tag{2.27}$$

This metric captures reliability across variability in initial conditions and uncertainties.

(ii) Time-to-handover. For successful cases, the time-to-handover is

$$t_{\text{HO}} \triangleq t_f - t_0, \tag{2.28}$$

with aggregate statistics such as mean, variance, and tail quantiles used to characterize typical and worst-case behavior across the scenario set.

(iii) Range and closing behavior. The evolution of $\rho(t)$ and $\dot{\rho}(t)$ provides insight into approach quality and detects stagnation or oscillatory behavior. In addition to terminal thresholds, monotonicity is not generally guaranteed under perturbations; however, boundedness and sustained progress can be assessed by windowed indicators such as

$$\Delta\rho_W(t) \triangleq \rho(t) - \rho(t - W), \tag{2.29}$$

for a chosen window length W (typically on the order of one or more orbital periods), or by verifying that $\rho(t)$ decreases on average over the maneuver. These indicators are useful for diagnosing failure modes even when terminal success is not achieved.

(iv) Control effort and propellant proxies. For low-thrust systems, control authority is bounded and propellant consumption is critical. A standard proxy for maneuver cost is the accumulated acceleration ("equivalent Δv"):

$$\Delta v_{\text{eq}} \triangleq \int_{t_0}^{t_f} \|\mathbf{a}_c(t)\|\, dt, \tag{2.30}$$

where $\mathbf{a}_c(t)$ is the commanded acceleration actually applied by the propulsion subsystem. When a thrust and mass-flow model is included, propellant consumption may be computed via:

$$\dot{m}(t) = -\frac{T(t)}{I_{sp} g_0}\, u(t), \qquad m_{\text{prop}} = m(t_0) - m(t_f), \tag{2.31}$$

or by an equivalent form consistent with throttle modeling. In this thesis, both Δv_{eq} and (when available) mass consumption are used to quantify efficiency.

(v) Constraint compliance and feasibility. Guidance must respect operational limits such as bounded acceleration magnitude, pointing constraints, and duty-cycle constraints. A generic feasibility condition for low-thrust actuation can be expressed as

$$\|\mathbf{a}_c(t)\| \leq a_{\max}(t) \qquad \forall t \in [t_0, t_f]. \tag{2.32}$$

Constraint compliance can be quantified by maximum violation and integrated violation measures, e.g.

$$\epsilon_{\max} \triangleq \max_{t\in[t_0,t_f]} \max\{0, \|\mathbf{a}_c(t)\| - a_{\max}(t)\}, \tag{2.33}$$

$$\epsilon_{\text{int}} \triangleq \int_{t_0}^{t_f} \max\{0, \|\mathbf{a}_c(t)\| - a_{\max}(t)\}\, dt. \tag{2.34}$$

These metrics are essential to distinguish trajectories that are nominally convergent but operationally infeasible.

2.1.6 Objective Statement

While this thesis does not assume a specific optimal control formulation, the guidance task can be summarized as producing a feasible control history that drives the relative state into $\mathcal{X}_f$ while minimizing a mission-relevant cost. A generic cost functional can be expressed as:

$$J \triangleq \alpha_t\, t_{\text{HO}} + \alpha_v\, \Delta v_{\text{eq}} + \alpha_c\, P_{\text{viol}} + \alpha_s\, P_{\text{smooth}}, \tag{2.35}$$

where P_{viol} penalizes constraint violations, P_{smooth} penalizes excessive command variability, and $\alpha_\cdot$ are positive weights reflecting mission priorities. In practice, this cost is used as a conceptual reference to interpret performance trade-offs; the actual evaluation throughout the thesis is conducted via the metric set defined above.

2.1.7 Role of the Metrics in the Thesis Workflow

The definitions in this section establish a consistent language for the remainder of the thesis. Later chapters use the handover set $\mathcal{X}_f$ and the metric families (success, time, efficiency, compliance, robustness) to:

- define acceptance criteria for simulation runs and dataset generation;
- compare alternative guidance implementations fairly under identical scenarios;
- quantify the impact of uncertainties and perturbations on mission-level outcomes.

By anchoring the analysis to trajectory-level and feasibility-oriented metrics, the study remains aligned with the requirements of onboard autonomy and operational relevance.

2.2 Dynamic Models and Considered Perturbations

2.2.1 Reference Frames

In orbital mechanics, the definition of reference frames is not a matter of notation only: orbital elements, perturbation models, and guidance commands depend on the adopted frame and on the associated conventions. This thesis employs an Earth-centered inertial frame for propagation and guidance evaluation, and a local orbital frame to parameterize and interpret acceleration commands
The specific construction of the inertial-to-Earth-fixed transformation is consistent with standard terrestrial and celestial reference-system conventions.[10]

ECI and ICRF

Let $\mathcal{F}_I \equiv \{\hat{\mathbf{i}}_I, \hat{\mathbf{j}}_I, \hat{\mathbf{k}}_I\}$ denote an ECI frame (ICRF/J2000-like). The origin is located at the Earth's center of mass. Vectors expressed in $\mathcal{F}_I$ may be denoted with a superscript $(\cdot)^I$ when needed (e.g., $\mathbf{r}^I, \mathbf{v}^I$).

The inertial state of a spacecraft is

$$\mathbf{x}^I(t) \triangleq \begin{bmatrix} \mathbf{r}^I(t) \\ \mathbf{v}^I(t) \end{bmatrix} \in \mathbb{R}^6, \tag{2.36}$$

where $\mathbf{r}^I$ is measured from the Earth center and $\mathbf{v}^I$ is the inertial velocity.

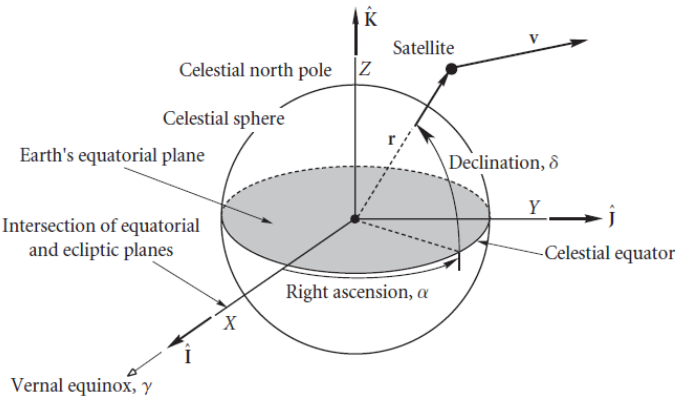

Figure 2.1: Earth-centered inertial frame $\mathcal{F}_I$ (ECI/ICRF)

ECEF and Inertial-to-Fixed Mapping (when required)

Some environmental models are naturally expressed in an Earth-fixed frame F_E rotating with the Earth (e.g., gravity harmonics aligned with the Earth's figure,

atmospheric drag, ground-relative quantities). When such models are enabled, inertial position vectors are mapped into ECEF through a time-dependent DCM $C_{E\leftarrow I}(t)$:

$$r^E(t) = C_{E\leftarrow I}(t)\, r^I(t). \tag{2.37}$$

The reverse mapping is $r^I = C_{E\leftarrow I}^{\top} r^E$ since $C_{E\leftarrow I}$ is orthonormal. Velocity transformations require additional kinematic terms due to the rotation of F_E and are discussed below. The specific construction of $C_{E\leftarrow I}(t)$ (Earth rotation angle, precession–nutation, polar motion, Earth orientation parameters) belongs to the simulation infrastructure; in this chapter we introduce only the notation and the frame roles.

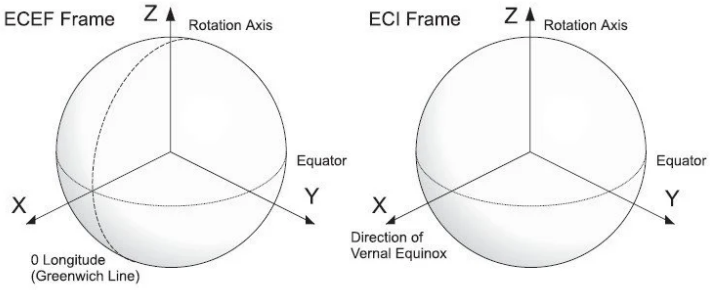

Figure 2.2: Relationship between inertial frame $\mathcal{F}_I$ and Earth-fixed frame $\mathcal{F}_E$ through the DCM $\mathbf{C}_{E\leftarrow I}(t)$.

Local Orbital Frame RTN (Command and Interpretation)

For guidance command parameterization and for an intuitive decomposition of accelerations, we adopt the local orbital frame $\mathcal{F}_R \equiv \{\hat{\mathbf{e}}_r, \hat{\mathbf{e}}_t, \hat{\mathbf{e}}_h\}$ (commonly RTN/RSW), built from the chaser instantaneous inertial state $(\mathbf{r}_c^I, \mathbf{v}_c^I)$.

The associated unit vectors are defined as:

$$\hat{\mathbf{e}}_r \triangleq \frac{\mathbf{r}_c^I}{\|\mathbf{r}_c^I\|}, \qquad \hat{\mathbf{e}}_h \triangleq \frac{\mathbf{r}_c^I \times \mathbf{v}_c^I}{\|\mathbf{r}_c^I \times \mathbf{v}_c^I\|}, \qquad \hat{\mathbf{e}}_t \triangleq \hat{\mathbf{e}}_h \times \hat{\mathbf{e}}_r, \tag{2.38}$$

where:

- $\hat{\mathbf{e}}_r$ is the **radial** unit vector (from Earth center to spacecraft),
- $\hat{\mathbf{e}}_t$ is the **along-track** (transverse) unit vector, tangent to the orbit,
- $\hat{\mathbf{e}}_h$ is the **cross-track** unit vector, normal to the orbital plane.

By construction, $\{\hat{\mathbf{e}}_r, \hat{\mathbf{e}}_t, \hat{\mathbf{e}}_h\}$ form a right-handed orthonormal triad.

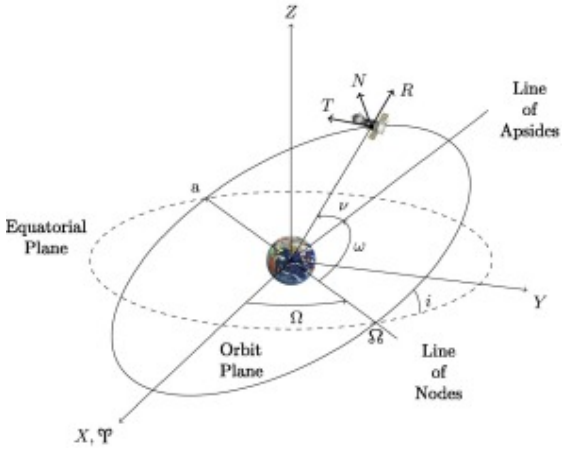

Figure 2.3: Definition of the RTN local orbital frame $\mathcal{F}_R$ from the chaser state $(\mathbf{r}_c, \mathbf{v}_c)$. The unit vectors are radial $(\hat{\mathbf{e}}_r)$, along-track $(\hat{\mathbf{e}}_t)$, and cross-track $(\hat{\mathbf{e}}_h)$.

The DCM mapping RTN components into inertial coordinates is:

$$\mathbf{C}_{I\leftarrow R} \triangleq \begin{bmatrix} \hat{\mathbf{e}}_r & \hat{\mathbf{e}}_t & \hat{\mathbf{e}}_h \end{bmatrix} \in \mathbb{R}^{3\times 3}, \qquad \mathbf{x}^I = \mathbf{C}_{I\leftarrow R}\, \mathbf{x}^R, \tag{2.39}$$

and conversely $\mathbf{x}^R = \mathbf{C}_{I\leftarrow R}^{\top} \mathbf{x}^I$.

In this work, the commanded low-thrust acceleration is naturally expressed in RTN components:

$$\mathbf{u}^R(t) \triangleq \begin{bmatrix} u_r(t) \\ u_t(t) \\ u_h(t) \end{bmatrix}, \qquad \mathbf{a}_c^I(t) = \mathbf{C}_{I\leftarrow R}(t)\, \mathbf{u}^R(t). \tag{2.40}$$

This parameterization directly separates the physical effects of thrust: radial (energy/shape), along-track (energy/phasing), and cross-track (plane change).

Remark on velocity transformations. While positions can be mapped as $\mathbf{r}^E = \mathbf{C}_{E\leftarrow I}\mathbf{r}^I$, the mapping of velocities between inertial and rotating frames requires additional kinematic terms due to the rotation of $\mathcal{F}_E$. Let $\boldsymbol{\omega}_E$ denote the Earth angular velocity expressed in $\mathcal{F}_E$. Then, the inertial and rotating-frame velocities satisfy

$$\left(\frac{d\mathbf{r}}{dt}\right)^I = \mathbf{C}_{I\leftarrow E}\left(\frac{d\mathbf{r}}{dt}\right)^E + \mathbf{C}_{I\leftarrow E}\left(\boldsymbol{\omega}_E \times \mathbf{r}^E\right), \tag{2.41}$$

or equivalently, in $\mathcal{F}_E$,

$$\mathbf{v}^E = \mathbf{C}_{E\leftarrow I}\mathbf{v}^I - \boldsymbol{\omega}_E \times \mathbf{r}^E. \tag{2.42}$$

In this thesis, these kinematic relations are handled consistently within the simulation infrastructure whenever Earth-fixed models (drag, harmonics, ground-relative quantities) are enabled. The purpose of this section is to define the notation and avoid ambiguity in later model descriptions.

Time Scales and Earth Orientation Parameters

When an ECI–ECEF mapping is required, it depends on the adopted time scales and Earth orientation parameters (EOP). In this thesis, the mapping is represented compactly by $\mathbf{C}_{E\leftarrow I}(t)$, where t denotes the simulation time argument. All details regarding time scales (e.g., UTC, UT1, TAI, TT) and EOP handling (polar motion, Earth rotation angle, precession–nutation) are encapsulated in the propagator implementation. This chapter introduces only the convention that $\mathbf{C}_{E\leftarrow I}(t)$ is orthonormal and continuously time-varying, and that it is applied consistently across all environment models and test cases.

2.2.2 Gauss variational equations: mapping from RTN acceleration to orbital-element rates

To explicitly relate low-thrust guidance commands to the evolution of the osculating orbit, we adopt Gauss' variational equations. Let the osculating Classical Orbital Elements be

$$\mathbf{y}_{\mathrm{COE}} \triangleq [a,\, e,\, i,\, \Omega,\, \omega,\, \nu]^{\top},$$

and decompose the specific perturbing acceleration in the local orbital frame $\{\hat{\mathbf{R}}, \hat{\mathbf{T}}, \hat{\mathbf{N}}\}$ as

$$\mathbf{a}_p \triangleq a_R\, \hat{\mathbf{R}} + a_T\, \hat{\mathbf{T}} + a_N\, \hat{\mathbf{N}}.$$

Define the semilatus rectum $p \triangleq a(1-e^2)$, the orbital radius $r \triangleq \frac{p}{1+e\cos\nu}$, and the specific angular momentum magnitude $h \triangleq \sqrt{\mu p}$. Moreover, let $u \triangleq \omega + \nu$ denote the argument of latitude. Under the osculating-element assumption, the element

rates are given by

$$\dot{a} = \frac{2a^2}{h}\Big(e\sin\nu\, a_R + \frac{p}{r}\, a_T\Big), \tag{2.43}$$

$$\dot{e} = \frac{1}{h}\Big(p\sin\nu\, a_R + \Big((p+r)\cos\nu + re\Big)\, a_T\Big), \tag{2.44}$$

$$\dot{i} = \frac{r\cos u}{h}\, a_N, \tag{2.45}$$

$$\dot{\Omega} = \frac{r\sin u}{h\sin i}\, a_N, \tag{2.46}$$

$$\dot{\omega} = \frac{1}{he}\Big(-p\cos\nu\, a_R + (p+r)\sin\nu\, a_T\Big) - \frac{r\sin u\cos i}{h\sin i}\, a_N, \tag{2.47}$$

$$\dot{\nu} = \frac{h}{r^2} + \frac{1}{he}\Big(p\cos\nu\, a_R - (p+r)\sin\nu\, a_T\Big). \tag{2.48}$$

Equations (2.43)–(2.48) highlight a control-affine structure that will be exploited later for guidance design and learning:

$$\dot{\mathbf{y}}_{\mathrm{COE}} = \mathbf{f}_0(\mathbf{y}) + \mathbf{B}(\mathbf{y})\,\mathbf{a}_{\mathrm{RTN}}, \qquad \mathbf{a}_{\mathrm{RTN}} \triangleq [a_R,\, a_T,\, a_N]^\top, \tag{2.49}$$

where $\mathbf{f}_0(\mathbf{y})$ collects the Keplerian contribution (e.g., the term h/r^2 in $\dot{\nu}$) and $\mathbf{B}(\mathbf{y})$ is the state-dependent mapping from RTN acceleration components to orbital-element rates. This mapping provides a direct physical interpretation of how acceleration direction affects size/shape/plane/phase evolution, and it will be referenced in the subsequent expert-law derivation and in the definition of learning targets.

2.2.3 Full Acceleration Model

The translational dynamics of chaser and target are propagated in an Earth-centered inertial frame $\mathcal{F}_I$ by numerical integration of the Cartesian state. For a generic spacecraft, the equations of motion are written as

$$\dot{\mathbf{r}}^I = \mathbf{v}^I, \tag{2.50}$$

$$\dot{\mathbf{v}}^I = \mathbf{a}^I_{2B}(\mathbf{r}^I) + \mathbf{a}^I_{J2}(\mathbf{r}^I) + \sum_{b\in\{\text{Sun, Moon}\}} \mathbf{a}^I_{3B,b}(\mathbf{r}^I, t) + \mathbf{a}^I_D(\mathbf{r}^I, \mathbf{v}^I, t) + \mathbf{a}^I_{SRP}(\mathbf{r}^I, t) + \mathbf{a}^I_c(t), \tag{2.51}$$

where $\mathbf{a}^I_c(t)$ is the commanded low-thrust acceleration (Section 2.2.4) and the remaining terms collect the considered environmental perturbations. The same model structure is used for both chaser and target; the target is treated as unactuated, i.e., $\mathbf{a}^I_c \equiv \mathbf{0}$.

Modularity and model levels. The acceleration model in (2.51) is implemented in modular form to enable controlled model-fidelity studies. Three fidelity levels are used throughout the thesis:

- **F0 (baseline):** two-body gravity + low-thrust control.
- **F1 (LEO-secular):** F0 + J_2 (and optional simplified drag).
- **F2 (enhanced):** F1 + third bodies (Sun/Moon) + SRP + higher-fidelity drag and actuation uncertainties.

This structure enables isolating the effect of each perturbation family on far-range convergence and on the learned policy generalization.

2.2.4 Continuous-Time Dynamics with Low-Thrust Control

The chaser translational dynamics are modeled as a controlled system:

$$\dot{\mathbf{r}}_c = \mathbf{v}_c, \tag{2.52}$$

$$\dot{\mathbf{v}}_c = \mathbf{a}_g(\mathbf{r}_c, t) + \mathbf{a}_p(\mathbf{r}_c, \mathbf{v}_c, t) + \mathbf{a}_c(t), \tag{2.53}$$

where $\mathbf{a}_g$ is the central-body gravitational acceleration, $\mathbf{a}_p$ collects perturbing accelerations, and $\mathbf{a}_c(t)$ is the commanded acceleration due to thrust.

The target dynamics are assumed known or predictable over the guidance horizon:

$$\dot{\mathbf{r}}_t = \mathbf{v}_t, \tag{2.54}$$

$$\dot{\mathbf{v}}_t = \mathbf{a}_g(\mathbf{r}_t, t) + \mathbf{a}_p(\mathbf{r}_t, \mathbf{v}_t, t), \tag{2.55}$$

which is equivalent to assuming either a cooperative target with broadcast ephemeris or a sufficiently accurate onboard prediction for guidance purposes.

For completeness, the nominal two-body gravity term can be written as

$$\mathbf{a}_g(\mathbf{r}) = -\mu \frac{\mathbf{r}}{\|\mathbf{r}\|^3}, \tag{2.56}$$

where μ is the Earth gravitational parameter.

For low-thrust propulsion, mass variation may be relevant. When included, the mass dynamics can be modeled as:

$$\dot{m}(t) = -\frac{T}{I_{sp} g_0}\, u(t), \tag{2.57}$$

where T is maximum thrust, I_{sp} is specific impulse, g_0 is standard gravity, and $u(t) \in [0,1]$ is a throttle factor. The commanded acceleration then becomes:

$$\mathbf{a}_c(t) = \frac{T}{m(t)}\, u(t)\, \hat{\mathbf{d}}(t), \qquad \|\hat{\mathbf{d}}(t)\| = 1, \tag{2.58}$$

with $\hat{\mathbf{d}}(t)$ a unit vector defining thrust direction.

2.2.5 Baseline Model and Perturbation Set

A baseline dynamic model is required for systematic analysis and repeatable simulation campaigns. The baseline typically includes two-body gravity (2.56) and low-thrust control (2.58), optionally augmented with simple perturbations. Robustness is then assessed by introducing additional effects or uncertainties.

In this thesis, perturbations are treated modularly and may include:

- **Non-spherical gravity:** low-degree harmonics (e.g., J_2) or higher-fidelity spherical harmonics when required. The main impact at far-range is the secular drift of orbital-plane and argument parameters over multiple revolutions, which indirectly affects phasing and handover readiness.
- **Third-body gravity:** Sun/Moon perturbations depending on orbit regime and maneuver duration.
- **Atmospheric drag:** when applicable (LEO cases), with uncertainty in density models and attitude-dependent drag area.
- **Solar radiation pressure:** as a bounded disturbance with parameter uncertainty (reflectivity coefficient, area-to-mass).
- **Actuation uncertainty:** thrust magnitude bias, misalignment, throttle quantization, and time delays.

All considered models are implemented within a closed-loop simulation environment using a Simulink-based orbital propagator. The propagator numerically integrates the nonlinear dynamics for chaser and target and applies the commanded acceleration input subject to the feasibility constraints in Section 2.3. This allows consistent evaluation of guidance behavior over long horizons and under progressively richer perturbation sets.

Two-Body and J_2 Perturbation

The central (two-body) gravitational acceleration is

$$\mathbf{a}_{2B}^{I}(\mathbf{r}) = -\mu \frac{\mathbf{r}}{\|\mathbf{r}\|^3}. \tag{2.59}$$

To capture the dominant effect of Earth oblateness in LEO, the J_2 perturbation is included. Let $\mathbf{r}$ be expressed in an Earth-fixed frame $\mathcal{F}_E$ with $\hat{\mathbf{k}}_E$ aligned with the Earth's rotation axis. Denoting $\mathbf{r}^E = [x\ y\ z]^\top$ and $r = \|\mathbf{r}^E\|$, the J_2 acceleration in $\mathcal{F}_E$ is

$$\mathbf{a}_{J2}^{E} = \frac{3J_2\mu R_E^2}{2r^5} \begin{bmatrix} x\left(5\frac{z^2}{r^2} - 1\right) \\ y\left(5\frac{z^2}{r^2} - 1\right) \\ z\left(5\frac{z^2}{r^2} - 3\right) \end{bmatrix}. \tag{2.60}$$

The inertial contribution is obtained by mapping $\mathbf{a}_{J_2}^E$ back to $\mathcal{F}_I$ through the DCM $\mathbf{C}_{I\leftarrow E}(t) = \mathbf{C}_{E\leftarrow I}^\top(t)$ introduced in Section 2.2.1.

Third-Body Gravitational Perturbations (Sun and Moon)

Third-body perturbations due to Sun and Moon are modeled as differential accelerations with respect to the Earth-centered frame. Let $\mathbf{r}$ be the spacecraft position w.r.t. Earth and let $\mathbf{r}_b(t)$ be the position of the third body b w.r.t. Earth, both expressed in $\mathcal{F}_I$. Then the third-body acceleration is

$$\mathbf{a}_{3B,b}^I = \mu_b \left(\frac{\mathbf{r}_b(t) - \mathbf{r}}{\|\mathbf{r}_b(t) - \mathbf{r}\|^3} - \frac{\mathbf{r}_b(t)}{\|\mathbf{r}_b(t)\|^3} \right), \tag{2.61}$$

where μ_b is the gravitational parameter of body b. This formulation ensures that the perturbation vanishes at the origin and is consistent with an Earth-centered inertial propagation.

Atmospheric Drag

In LEO, atmospheric drag is modeled as an acceleration opposite to the velocity relative to the atmosphere. Let $\mathbf{v}^E$ be the spacecraft velocity expressed in $\mathcal{F}_E$ and let $\boldsymbol{\omega}_E$ be Earth's rotation vector. The velocity relative to the co-rotating atmosphere is approximated as

$$\mathbf{v}_{rel}^E = \mathbf{v}^E - \boldsymbol{\omega}_E \times \mathbf{r}^E. \tag{2.62}$$

The drag acceleration is

$$\mathbf{a}_D^E = -\frac{1}{2} \frac{C_D A}{m} \rho_{atm}(h,t) \, \|\mathbf{v}_{rel}^E\| \, \mathbf{v}_{rel}^E, \tag{2.63}$$

where C_D is the drag coefficient, A is the reference area, m is the spacecraft mass, and ρ_{atm} is the atmospheric density model as a function of altitude h (and possibly time/solar activity). The inertial contribution is obtained through the ECEF-to-ECI mapping.

Solar Radiation Pressure

Solar radiation pressure (SRP) is modeled as a bounded disturbance directed approximately away from the Sun. Let $\mathbf{r}_\odot(t)$ be the Sun position w.r.t. Earth and let $\mathbf{r}_{s/c}$ be the spacecraft position. Define the Sun-to-spacecraft direction

$$\hat{\mathbf{s}} = \frac{\mathbf{r}_{s/c} - \mathbf{r}_\odot}{\|\mathbf{r}_{s/c} - \mathbf{r}_\odot\|}. \tag{2.64}$$

A common acceleration model is

$$\mathbf{a}_{SRP}^{I} = -P_{\odot}\,\frac{C_R A}{m}\,\hat{\mathbf{s}}\,\eta(t), \tag{2.65}$$

where $P_{\odot}$ is the solar radiation pressure at 1 AU (scaled appropriately if desired), C_R is a reflectivity coefficient, and $\eta(t) \in [0,1]$ is a shadowing factor accounting for eclipse conditions.

2.3 Operational Constraints and Safety Requirements

2.3.1 Actuation Limits

Low-thrust rendezvous is fundamentally constrained by limited control authority. The primary hard constraint is the bound on the commanded acceleration magnitude:

$$\|\mathbf{a}_c(t)\| \le a_{\max}(t), \tag{2.66}$$

where $a_{\max}(t)$ may vary with mass as implied by (2.58). In practice, guidance laws (expert and learned) may output a *requested* acceleration $\mathbf{a}_{\text{req}}(t)$ which is then mapped into a feasible command by saturation and operational gating:

$$\mathbf{a}_c(t) \triangleq \Pi_{\mathcal{U}(t)}\Big(\mathbf{a}_{\text{req}}(t)\Big), \tag{2.67}$$

where $\Pi_{\mathcal{U}(t)}(\cdot)$ denotes the projection (or clipping) onto the set of admissible accelerations $\mathcal{U}(t)$ at time t. This representation is convenient because it decouples the policy definition from the actuation implementation while guaranteeing feasibility by construction whenever $\mathcal{U}(t)$ is non-empty.

Additional operational constraints are captured through the definition of $\mathcal{U}(t)$ and may include:

- **Thrust pointing constraints:** the thrust direction $\hat{\mathbf{d}}(t)$ restricted by attitude constraints, keep-out cones, or Sun-safe/thermal pointing envelopes. A generic representation is
$$\hat{\mathbf{d}}(t) \in \mathcal{D}(t) \subset \mathbb{S}^2, \tag{2.68}$$
where $\mathcal{D}(t)$ is a time-varying admissible direction set.

- **Duty-cycle / availability constraints:** limitations on continuous firing duration, minimum on/off times, or power-limited thrusting (e.g., eclipse). These can be modeled by a scalar availability factor $\gamma(t) \in [0,1]$ such that
$$0 \le u(t) \le \gamma(t). \tag{2.69}$$

- **Rate and smoothness constraints:** bounds on direction and throttle rates, e.g.,

$$\|\dot{\hat{\mathbf{d}}}(t)\| \leq \dot{d}_{\max}, \qquad |\dot{u}(t)| \leq \dot{u}_{\max}, \tag{2.70}$$

  which prevent unrealistically fast switching and reflect actuator and attitude-control limitations.

These constraints ensure that the generated guidance is compatible with realistic propulsion and attitude-control subsystems, and they provide a consistent basis for feasibility assessment.

2.3.2 Mission and Safety Constraints

Even in far-range, safety-driven constraints influence guidance design because long-duration maneuvers can inadvertently lead to undesired approach geometries or excessive closing rates. In this thesis, safety is described through *state constraints* and *handover-compatibility constraints*.

Keep-out regions (state safety). Let $\mathbf{x}_{\text{rel}}(t)$ denote a chosen relative-state representation (e.g., inertial $(\boldsymbol{\rho}, \dot{\boldsymbol{\rho}})$, or a target-centered LVLH representation). Define an *unsafe* set $\mathcal{K}$ and the corresponding safe set $\mathcal{S}$ as

$$\mathcal{S} \triangleq \mathbb{R}^n \setminus \mathcal{K}. \tag{2.71}$$

A generic safety requirement is

$$\mathbf{x}_{\text{rel}}(t) \in \mathcal{S} \qquad \forall t \in [t_0, t_f]. \tag{2.72}$$

Typical examples of $\mathcal{K}$ include exclusion spheres/cylinders, forbidden line-of-sight cones, or constraints that prevent approach from undesired directions. In this chapter, the precise geometry of $\mathcal{K}$ is kept generic; later chapters introduce specific forms and safety-enforcement mechanisms consistent with the adopted guidance architecture.

Geometric corridors and handover compatibility. In addition to avoiding unsafe states, far-range guidance must steer the system toward a configuration compatible with subsequent mid-/near-range logic. This is captured by the handover set $\mathcal{X}_f$ (Section 2.1.4), whose additional constraint $\phi(\cdot)$ can be used to encode an approach corridor, relative-orbit alignment proxies, or bounded relative-energy conditions. Conceptually, these constraints ensure that the far-range phase does not achieve terminal thresholds through a transient or operationally fragile configuration.

Bounded relative energy and closing speed. Beyond the terminal limits on $\rho(t)$ and $\|\dot{\boldsymbol{\rho}}(t)\|$, it is often useful to monitor progress and safety through energy-like or rate-based indicators (Section 2.4.2). In particular, excessive closing rates can compromise the feasibility of later capture logic, while sustained increases in separation indicate divergence and may trigger early termination in simulation.

Operational time constraints. A maximum allowed time-to-handover $t_{\max}$ is imposed to reflect mission timelines, power/thermal constraints, eclipse scheduling, or communication-window constraints. This is treated as a hard constraint in evaluation:

$$t_f - t_0 \leq t_{\max}. \tag{2.73}$$

In a verification-oriented context, safety is interpreted as maintaining bounded, predictable behavior under uncertainties, and ensuring that the guidance policy does not generate infeasible commands or unsafe trajectories that would require frequent ground intervention.

2.3.3 Problem Statement

Given initial conditions $(\mathbf{r}_c(t_0), \mathbf{v}_c(t_0), m(t_0))$ and target ephemeris $(\mathbf{r}_t(t), \mathbf{v}_t(t))$, the guidance problem consists of generating a feasible command history $\mathbf{a}_c(t)$ (or equivalently $u(t), \hat{\mathbf{d}}(t)$) such that the relative state reaches a desired terminal set while satisfying operational and safety constraints:

$$\begin{aligned} &\text{find } \mathbf{a}_c(t), \quad t \in [t_0, t_f], \\ &\text{s.t. dynamics (2.52)–(2.57)}, \\ &\quad \mathbf{a}_c(t) \in \mathcal{U}(t) \ \text{ (including (2.66) and additional limits)}, \\ &\quad \mathbf{x}_{\text{rel}}(t) \in \mathcal{S} \ \ \forall t \in [t_0, t_f] \ \text{ (safety)}, \\ &\quad (\boldsymbol{\rho}(t_f), \dot{\boldsymbol{\rho}}(t_f)) \in \mathcal{X}_f, \qquad t_f - t_0 \leq t_{\max}. \end{aligned} \tag{2.74}$$

The terminal set $\mathcal{X}_f$ encodes the far-to-mid/near handover requirements, while $\mathcal{U}(t)$ and $\mathcal{S}$ capture actuation feasibility and mission safety, respectively.

2.4 Convergence Criteria and Termination Conditions

2.4.1 Handover (Terminal) Conditions

Convergence is defined as reaching a state suitable for handover to subsequent rendezvous phases. A generic terminal condition can be expressed as:

$$\|\boldsymbol{\rho}(t_f)\| \leq \rho_{\text{tol}}, \qquad \|\dot{\boldsymbol{\rho}}(t_f)\| \leq v_{\text{tol}}, \tag{2.75}$$

possibly augmented with additional constraints on approach geometry or relative orbital elements (cf. (2.24)). The thresholds ρ_{tol} and v_{tol} are mission-dependent; in this thesis they are treated as design parameters and used consistently across all evaluations.

In long-horizon low-thrust rendezvous, it is often useful to require *persistence* of terminal satisfaction to avoid declaring success due to transient threshold crossings. This can be implemented by verifying that the inequalities in (2.75) hold continuously over a time window of duration T_{pers} prior to t_f.

2.4.2 Progress and Stagnation Detection

Because far-range low-thrust maneuvers can span long durations, it is important to detect stagnation or divergence early. In addition to the terminal condition, progress can be monitored through indicators such as:

- decreasing trend in $\|\boldsymbol{\rho}(t)\|$ over moving windows (e.g., (2.29));
- boundedness of relative speed and relative energy proxies (preventing sustained growth of separation and closing speed);
- sustained feasibility of commanded actions under constraints (Section 2.3).

These indicators support robust simulation campaign management and provide diagnostic information when guidance fails to converge.

2.4.3 Termination Conditions in Simulation

Each simulation run is terminated according to one of the following mutually exclusive outcomes:

- **Success (handover achieved):** the terminal condition is satisfied (Eq. (2.75) and any additional handover constraints) and, when enabled, it persists for at least a dwell time T_{pers} to avoid declaring success due to transient threshold crossings.
- **Timeout:** the maximum allowed duration is reached, i.e., $t - t_0 \geq t_{\max}$, without meeting the persistence-based success condition.
- **Operational infeasibility / constraint violation:** a persistent or severe violation of operational bounds occurs. This includes, for example, sustained acceleration demand beyond the admissible set $\mathcal{U}(t)$ (Section 2.3.1), excessive integrated violation (e.g., ϵ_{int}), or repeated command saturation indicating loss of effective control authority. Infeasibility is declared when violations persist for longer than a configured window T_{viol} or exceed a peak threshold.

- **Safety violation:** the relative state enters an unsafe set (keep-out region) $\mathcal{K}$, equivalently violating the safety requirement $\mathbf{x}_{\text{rel}}(t) \in \mathcal{S}$ (Section 2.3.2). This outcome captures cases where terminal convergence might still be possible, but the trajectory is operationally unacceptable.
- **Numerical failure:** the numerical propagation fails (integration error, NaN/Inf state) or the model becomes invalid due to implementation-domain issues (e.g., atmosphere model outside validity range).

For post-processing and failure-mode attribution, each run also records diagnostic indicators such as: minimum achieved range, maximum closing rate, peak and time-integrated command saturation/violation, total thrusting time, and geometry/phasing residuals ($e_{\text{geom}}, \Delta\lambda$) at termination. This classification enables consistent success-rate analysis and supports interpretation of results in terms of mission feasibility and robustness.

2.4.4 Implications for Later Chapters

The formulation presented in this chapter establishes the reference problem for the development of autonomous guidance policies. Later chapters build on this foundation by introducing a baseline guidance law, generating representative simulation data over the defined operational envelope, and evaluating alternative guidance implementations using the metrics and termination logic defined here.

Chapter 3

Expert Guidance Law: A Q-law-Inspired Hybrid Strategy

This chapter presents the *expert guidance policy* adopted in this thesis to autonomously generate far-range, low-thrust rendezvous trajectories and to provide supervision targets for the neural surrogate policy introduced in the following chapters. Although the term "Q-law" is retained throughout the manuscript for conciseness, the implemented expert is more precisely described as a *Q-law-inspired hybrid guidance law*. Its core guidance mechanism remains consistent with the Q-law philosophy, namely the construction of a local sensitivity-based descent direction in orbital-element space. However, this core mechanism is embedded within a broader supervisory architecture that explicitly accounts for bounded actuation, numerical conditioning, mode sequencing, and robust closed-loop behavior over long simulation horizons.[2, 3, 11]

The resulting expert combines two coordinated guidance channels:

- a **geometry-closure channel**, formulated in MEE, which generates a RTN acceleration command aligned with a weighted local descent direction;
- a **phase-correction channel**, which generates controlled tangential drift through a temporary semi-major-axis offset and is activated only once orbital geometry has been sufficiently aligned.

Their outputs are blended and then processed through feasibility enforcement, gating logic, and persistence-based handover detection.

This architecture is motivated by the role that the expert must play within the overall thesis workflow. The objective is not merely to obtain convergence

in a nominal scenario, but to define a guidance policy that remains numerically robust, bounded, repeatable, and diagnostically transparent across the large Monte Carlo campaigns used for dataset generation, training, and comparative validation. For this reason, implementation-level mechanisms such as tapering, deadbands, feasibility projection, and persistence timers are treated here as integral parts of the expert law rather than as secondary implementation details.

Remark on terminology ("Q-law-inspired"). Classical Q-law formulations are usually derived from the descent of a Lyapunov-like potential through an analytical sensitivity model, often under assumptions that support relatively clean theoretical guarantees at the guidance-law level. The expert implemented in this thesis preserves the essential operational principle of Q-law - namely, local descent in element space driven by control sensitivities, but augments it with supervisory mechanisms required for large-scale simulation repeatability and numerically stable closed-loop execution. Accordingly, the term *Q-law-inspired* is adopted throughout the manuscript, to distinguish the implemented law from classical analytical Q-law formulations while preserving its Lyapunov-guided orbital-element descent rationale.[2, 3, 11]

3.1 Purpose of the Expert Guidance within the Thesis

The expert guidance law occupies a central methodological role in this thesis. Its function is not limited to producing a single feasible trajectory; rather, it supports the entire expert-to-surrogate workflow. Specifically, it is used as:

- a **reference autonomous policy**, to solve the far-range low-thrust rendezvous problem under the assumptions and constraints introduced in Chapter 2;
- a **demonstration generator**, to produce state–action pairs for supervised imitation learning and subsequent dataset-refinement procedures;
- a **benchmark policy**, against which the closed-loop behavior of the learned surrogate can be compared using a deterministic and physically interpretable baseline.

This dual role - controller and data generator - imposes requirements that go beyond classical guidance design. In particular, the expert should satisfy:

- **feasibility**: commanded accelerations must remain compatible with the available thrust authority and should not introduce artificial numerical pathologies;

- **low-chatter behavior**: the control law should avoid unnecessary switching and residual micro-actuation near convergence;
- **interpretability and diagnosability**: intermediate metrics, gating decisions, and channel contributions should be explicit and loggable for post-run analysis;
- **repeatability**: for fixed initial conditions and tuning parameters, the policy should generate deterministic trajectories and actions suitable for learning.

These requirements motivate the hybrid structure described in the remainder of the chapter. The design target is therefore not a formally optimal law, but a reliable and physically meaningful expert whose outputs can be used both for closed-loop guidance and for the construction of consistent imitation-learning datasets.

3.2 Guidance Interface, Discrete-Time Update, and Command Outputs

The guidance law is evaluated at discrete update instants t_k, with fixed guidance period T_g, consistent with the simulation step and/or scheduling logic introduced in Chapter 2. At each update, the policy receives the inertial states of chaser and target,

$$(\mathbf{r}_c^I, \mathbf{v}_c^I), \qquad (\mathbf{r}_t^I, \mathbf{v}_t^I), \tag{3.1}$$

together with the available acceleration bound $a_{\max}(t_k)$, the gravitational parameter μ, and an enable/disable flag from higher-level supervisory logic.

The primary output is the commanded inertial acceleration for the chaser,

$$\mathbf{a}_c^I(t_k) \in \mathbb{R}^3, \tag{3.2}$$

along with a completion/handover flag indicating persistent satisfaction of the far-range terminal conditions. In addition, the expert exposes auxiliary diagnostic quantities such as e_{geom}, $\Delta\lambda_m$, gate flags, and channel norms. These signals do not alter the control command, but are useful for trajectory inspection, dataset curation, and failure diagnosis.

3.2.1 Internal RTN Formulation and Frame Mapping

Although the external interface is expressed in inertial coordinates, the internal guidance logic is formulated in the local orbital RTN frame of the chaser. The

RTN basis is defined as

$$\hat{\mathbf{R}} = \frac{\mathbf{r}_c^I}{\|\mathbf{r}_c^I\|}, \tag{3.3}$$

$$\hat{\mathbf{N}} = \frac{\mathbf{r}_c^I \times \mathbf{v}_c^I}{\|\mathbf{r}_c^I \times \mathbf{v}_c^I\|}, \tag{3.4}$$

$$\hat{\mathbf{T}} = \hat{\mathbf{N}} \times \hat{\mathbf{R}}, \tag{3.5}$$

with associated rotation matrix

$$C_{I \leftarrow R} = \begin{bmatrix} \hat{\mathbf{R}} & \hat{\mathbf{T}} & \hat{\mathbf{N}} \end{bmatrix}. \tag{3.6}$$

Let

$$\mathbf{u}^R(t_k) = \begin{bmatrix} u_R(t_k) \\ u_T(t_k) \\ u_N(t_k) \end{bmatrix} \tag{3.7}$$

denote the commanded acceleration in RTN coordinates. The inertial command is obtained as

$$\mathbf{a}_c^I(t_k) = C_{I \leftarrow R}(t_k)\, \mathbf{u}^R(t_k). \tag{3.8}$$

This internal decomposition is particularly natural for low-thrust rendezvous because the three RTN directions are associated with distinct dynamical effects. The tangential component primarily affects orbital energy and semi-major-axis evolution, the normal component drives plane alignment, and the radial component contributes to shape and orientation corrections. Expressing the expert in RTN coordinates therefore facilitates both physical interpretation and structured command synthesis.[7]

3.2.2 Element Conversion and MEE Internal State

The Cartesian states are converted into orbital elements and subsequently mapped into Modified Equinoctial Elements,

$$\mathbf{x}_{\text{mee}} = [p, f, g, h, k, L]^\top, \tag{3.9}$$

for both chaser and target. The MEE representation is adopted because it avoids the classical singularities associated with near-circular and near-equatorial orbits while remaining well suited to low-thrust element-rate formulations.[6]

The chaser–target error in MEE coordinates is defined as

$$\Delta\mathbf{x}_{\text{mee}} = \mathbf{x}_{\text{mee},c} - \mathbf{x}_{\text{mee},t} = [\Delta p, \Delta f, \Delta g, \Delta h, \Delta k, \Delta L]^\top. \tag{3.10}$$

All periodic quantities are wrapped consistently, for instance $\Delta L \in (-\pi, \pi]$, in order to avoid discontinuities at angular branch cuts.

3.3 Hybrid Guidance Architecture and Supervisory Logic

The expert is organized as a hybrid supervisor with two partially decoupled objectives:

- **orbit-geometry matching**, aimed at reducing mismatch in size, shape, and orbital plane;
- **phasing correction**, aimed at reducing along-track timing mismatch once the geometry has approached the target manifold.

This separation is not merely convenient from an implementation perspective; it reflects the multiscale character of the far-range low-thrust rendezvous problem. Plane and shape corrections often require long cumulative action, whereas aggressive phasing too early in the trajectory may create inefficient spiraling, competing objectives, or stalled convergence. In other words, attempting to regulate timing mismatch before the chaser has reached an appropriate orbital family can degrade overall guidance quality. The expert therefore treats geometry closure as the primary task and enables the phase channel only when a geometry-ready condition is satisfied.

Figure 3.1 summarizes the conceptual structure of the implemented expert.

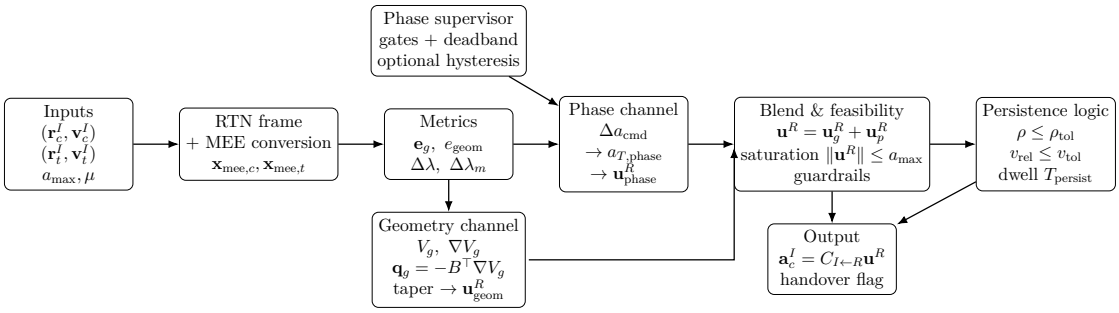

Figure 3.1: Conceptual architecture of the Q-law-inspired hybrid expert guidance. The expert combines a geometry-closure channel based on local MEE sensitivities with a gated phase-correction channel based on tangential drift, and augments both with feasibility enforcement and persistence-based handover logic.

Operationally, each guidance update consists of the following conceptual stages: construction of geometry and phase error metrics; computation of the geometry descent direction; evaluation of phase-readiness conditions; synthesis and saturation of the combined RTN command; optional guardrail enforcement; and update of the persistence-based handover logic. The detailed formulation of each stage is presented next.

3.4 Geometry-Closure Channel in MEE Space

3.4.1 Geometry Error Vector and Normalized Metric

The geometry channel excludes the phase variable and focuses on the five-element vector

$$\mathbf{e}_g \triangleq [\Delta p, \Delta f, \Delta g, \Delta h, \Delta k]^\top. \tag{3.11}$$

A normalized scalar indicator is then introduced as

$$e_{\text{geom}} \triangleq \sqrt{\left(\frac{\Delta p}{p_{\text{ref}}}\right)^2 + (\Delta f)^2 + (\Delta g)^2 + (\Delta h)^2 + (\Delta k)^2}, \tag{3.12}$$

where p_{ref} is a positive scaling reference, typically chosen as $p_{\text{ref}} = \max(p_t, p_{\min})$ with a small floor $p_{\min} > 0$.

The scalar e_{geom} is used for task sequencing, tapering, and gating. It is not presented as a strict Lyapunov function for the full nonlinear closed-loop dynamics; rather, it is a normalized geometry indicator that captures, in compact form, how far the chaser remains from the target orbit family in the variables most relevant to far-range closure.

3.4.2 Weighted Quadratic Potential

The local geometry objective is encoded through the weighted quadratic potential

$$V_g(\mathbf{e}_g) = \frac{1}{2}\mathbf{e}_g^\top W_g \mathbf{e}_g, \tag{3.13}$$

where $W_g = \text{diag}(w_p, w_f, w_g, w_h, w_k)$ is diagonal and positive definite.

The diagonal weights assign relative priority to:

- **size mismatch** through Δp;
- **eccentricity-vector mismatch** through Δf and Δg;
- **plane mismatch** through Δh and Δk.

This weighting is important because low-thrust rendezvous is intrinsically anisotropic: depending on the initial geometry, plane correction may dominate feasibility, whereas in other cases shape and size alignment may be the limiting factor. The weights therefore serve as a compact design mechanism to encode guidance priorities.

The gradient of the potential with respect to the full MEE state is written as

$$\nabla V_g = \begin{bmatrix} W_g \mathbf{e}_g \\ 0 \end{bmatrix}, \tag{3.14}$$

where the final zero explicitly suppresses direct phase minimization within the geometry channel.

3.4.3 Control-Affine MEE Dynamics and Sensitivity Matrix

Under the standard osculating-element approximation, the MEE dynamics can be written in control-affine form as

$$\dot{\mathbf{x}}_{\mathrm{mee}} = \mathbf{f}_0(\mathbf{x}_{\mathrm{mee}}, t) + B_{\mathrm{mee}}(\mathbf{x}_{\mathrm{mee}})\, \mathbf{a}_{RTN}, \tag{3.15}$$

where $\mathbf{f}_0$ collects the uncontrolled drift terms and B_{mee} maps RTN acceleration into MEE element rates.[6] In this work, the expert uses the standard two-body B_{mee} associated with the Gauss variational equations in MEE form, while modeled perturbations are accounted for in the simulation environment.

Define the auxiliary quantities

$$w = 1 + f\cos L + g\sin L, \qquad s^2 = 1 + h^2 + k^2, \qquad \beta = \sqrt{\frac{p}{\mu}}. \tag{3.16}$$

The acceleration-dependent component of the geometry-relevant MEE dynamics is then

$$\dot{p} = \frac{2p\beta}{w}\, a_T, \tag{3.17}$$

$$\dot{f} = \beta\left[\sin L\, a_R + \frac{(w+1)\cos L + f}{w}\, a_T - \frac{h\sin L - k\cos L}{w}\, a_N\right], \tag{3.18}$$

$$\dot{g} = \beta\left[-\cos L\, a_R + \frac{(w+1)\sin L + g}{w}\, a_T + \frac{h\cos L + k\sin L}{w}\, a_N\right], \tag{3.19}$$

$$\dot{h} = \beta\left(\frac{s^2}{2w}\cos L\right) a_N, \tag{3.20}$$

$$\dot{k} = \beta\left(\frac{s^2}{2w}\sin L\right) a_N. \tag{3.21}$$

Numerical safeguards. For the near-circular regimes of interest, the expressions above are generally well conditioned. Nevertheless, when w becomes small, the element-rate coefficients can be artificially amplified. The implementation therefore enforces a lower bound $w \geq w_{\min}$ in intermediate computations, or equivalently attenuates the associated command when w approaches this limit. This does not alter the conceptual guidance structure, but avoids rare numerical spikes that would otherwise contaminate long rollouts.

3.4.4 Q-law-Inspired Local Descent Direction in RTN

The local first-order variation of the geometry potential is

$$\dot{V}_g = \nabla V_g^{\top}\dot{\mathbf{x}}_{\mathrm{mee}} = \nabla V_g^{\top}\mathbf{f}_0 + \nabla V_g^{\top} B_{\mathrm{mee}}\mathbf{a}_{RTN}. \tag{3.22}$$

Since the control affects V_g through the second term, a natural local descent direction in acceleration space is

$$\mathbf{q}_g \triangleq -B_{\text{mee}}^{\top} \nabla V_g. \tag{3.23}$$

Intuitively, $\mathbf{q}_g$ identifies the RTN acceleration direction that most strongly decreases the linearized geometry potential under a unit-norm acceleration constraint.

If $\|\mathbf{q}_g\| > \varepsilon_q$, the normalized geometry direction is defined as

$$\hat{\mathbf{q}}_g = \frac{\mathbf{q}_g}{\|\mathbf{q}_g\|}. \tag{3.24}$$

Otherwise, the geometry command is set to zero at that update. This avoids amplifying numerical noise when the local sensitivity direction becomes too small or poorly conditioned to be interpreted reliably.

3.4.5 Thrust Tapering and Predicted-Descent Consistency Check

Driving the geometry channel at full authority at every update is unnecessary and potentially undesirable near convergence. To regularize the command, the expert introduces a smooth tapering law based on the scalar indicator e_{geom}:

$$r_g = \text{sat}_{[0,1]}\left(\frac{e_{\text{geom}}}{e_{\text{quiet}}}\right), \qquad a_{g,\text{cmd}} = a_{\max}\, r_g^{\gamma}, \tag{3.25}$$

where e_{quiet} determines the onset of progressive coasting and $\gamma \geq 1$ shapes the tapering profile. In this work, $\gamma = 2$ is a representative choice. The resulting geometry command is

$$\mathbf{u}_{\text{geom}}^{R} = a_{g,\text{cmd}}\, \hat{\mathbf{q}}_g. \tag{3.26}$$

A further robustness measure consists in evaluating the predicted linearized contribution of the geometry command:

$$\dot{V}_{g,\text{pred}} = \nabla V_g^{\top} B_{\text{mee}}\, \mathbf{u}_{\text{geom}}^{R}. \tag{3.27}$$

If $\dot{V}_{g,\text{pred}}$ is not sufficiently negative within a prescribed tolerance, the command can be attenuated or rejected for that update. This consistency check reduces the risk of spurious thrusting in locally ill-conditioned configurations and improves the regularity of expert rollouts used for dataset generation.

3.4.6 Implementation Safeguards for Numerical Robustness

In addition to the analytical structure outlined above, practical execution requires a small set of explicit numerical safeguards to ensure deterministic behavior over large simulation campaigns. The adopted measures include:

- **sensitivity conditioning**: intermediate quantities such as $w = 1 + f\cos L + g\sin L$ are lower-bounded to prevent coefficient amplification;
- **direction reliability**: if the norm of the descent direction falls below ε_q, the geometry command is suppressed for that update;
- **finite-value enforcement**: non-finite intermediate computations trigger a conservative fallback, typically a zero command, to prevent contamination of long rollouts and stored datasets.

These mechanisms are not intended as operational fault-management logic. Their purpose is narrower and specific to the thesis context: to guarantee that the expert remains numerically stable, repeatable, and suitable for simulation-driven learning.

3.5 Phase-Correction Channel via Tangential Drift

3.5.1 Motivation: Geometry Closure Does Not Guarantee Rendezvous Readiness

In far-range rendezvous, approaching the target orbital geometry does not automatically imply readiness for handover. The chaser may evolve toward the correct orbit family while remaining poorly phased, which translates into persistent along-track separation or very slow residual closure. This issue becomes more pronounced in low-thrust regimes, where the geometry channel naturally reduces its authority near convergence because of tapering.

For this reason, the expert includes a dedicated phase-correction channel that acts predominantly through tangential acceleration. The role of this channel is not to replace geometry guidance, but to exploit mean-motion mismatch in a controlled way once the dominant geometry discrepancies have been reduced. This sequencing is important: at long horizon and low thrust, phasing is most effective when it operates on top of an already aligned orbital geometry, rather than in competition with large plane or shape corrections.

3.5.2 Phase Variable Definition and Filtering

The phase mismatch is represented using a mean-longitude-like variable,

$$\lambda = \Omega + \omega + M, \tag{3.28}$$

and the wrapped difference

$$\Delta\lambda = \mathrm{wrap}_{\pi}(\lambda_c - \lambda_t). \tag{3.29}$$

Because osculating elements exhibit short-period oscillations, direct use of $\Delta\lambda$ can induce unnecessary switching in the phase command. The implementation therefore employs a filtered estimate, denoted $\Delta\lambda_m$, obtained through a low-pass mechanism with time scale selected to suppress short-period fluctuations while preserving the secular phase trend relevant to drift control.

3.5.3 Phase Activation Logic (Gating, Deadbands, Hysteresis)

The phase channel is activated only when the guidance is deemed *geometry-ready*. A typical activation logic combines:

- a **geometry gate**: $e_{\mathrm{geom}} \leq e_{\mathrm{gate}}$;
- a **plane-readiness condition**: for instance $\sqrt{\Delta h^2 + \Delta k^2} \leq e_{\mathrm{plane}}$;
- a **phase deadband**: phase actuation is disabled when $|\Delta\lambda_m| \leq \lambda_{\mathrm{db}}$;
- **optional hysteresis**: distinct on/off thresholds can be used to prevent rapid toggling near the activation boundary.

This supervisory logic plays an important role in maintaining clean and interpretable expert behavior. Without it, the phase channel could attempt to correct residual timing mismatch while the geometry channel is still engaged in dominant orbit-matching tasks, thereby reducing efficiency and increasing command variability.

3.5.4 Tangential Drift through Temporary Semi-Major Axis Offset

The phase channel is based on a simple physical principle: a temporary semi-major-axis offset produces a mean-motion mismatch, which in turn generates secular phase drift. Using the first-order relationship

$$n = \sqrt{\frac{\mu}{a^3}}, \qquad \Delta n \approx -\frac{3}{2} n \frac{\Delta a}{a}, \tag{3.30}$$

[7, 12] a desired phase correction over a characteristic time τ_{phase} can be translated into a commanded semi-major-axis offset

$$\Delta a_{\mathrm{cmd}} \approx \frac{2}{3}\, a\, \frac{\Delta\lambda_m}{n\, \tau_{\mathrm{phase}}}, \tag{3.31}$$

with sign conventions consistent with the implementation.

To prevent excessive excursions from the target orbit family, the commanded offset is bounded according to

$$|\Delta a_{\mathrm{cmd}}| \le \Delta a_{\max}, \qquad \frac{|\Delta a_{\mathrm{cmd}}|}{a} \le \eta_{a,\max}. \tag{3.32}$$

These limits preserve the interpretation of phasing as a temporary drift mechanism rather than a redefinition of the geometry objective.

3.5.5 Conversion to Tangential Acceleration Command

Let

$$a_{\mathrm{des}} = a_t + \Delta a_{\mathrm{cmd}}, \qquad e_a = a_{\mathrm{des}} - a_c. \tag{3.33}$$

The phase channel then acts as a servo on the semi-major-axis error e_a. Under near-circular assumptions, one may use the approximation

$$\dot{a} \approx \frac{2a^{3/2}}{\sqrt{\mu}}\, a_T. \tag{3.34}$$

Imposing a first-order decay law $\dot{e}_a \approx -e_a/\tau_a$ yields the raw tangential command

$$a_{T,\mathrm{raw}} \approx \frac{e_a}{\tau_a}\,\frac{\sqrt{\mu}}{2a^{3/2}}. \tag{3.35}$$

This command is subsequently shaped and bounded as

$$a_{T,\mathrm{phase}} = w_{\mathrm{phase}}\, w_\lambda\, \mathrm{clamp}(a_{T,\mathrm{raw}}, -A_{\mathrm{phase,max}}, A_{\mathrm{phase,max}})\,, \tag{3.36}$$

where the weights encode readiness and phase-error scaling. The phase channel acts only along the tangential direction, i.e.

$$\mathbf{u}^R_{\mathrm{phase}} = \begin{bmatrix} 0 \\ a_{T,\mathrm{phase}} \\ 0 \end{bmatrix}. \tag{3.37}$$

3.6 Command Blending, Feasibility Enforcement, and Guardrails

3.6.1 Hybrid Command Synthesis

The preliminary RTN command is obtained by summing the contributions of the two channels:

$$\mathbf{u}^R_{\mathrm{raw}} = \mathbf{u}^R_{\mathrm{geom}} + \mathbf{u}^R_{\mathrm{phase}}. \tag{3.38}$$

This additive structure preserves interpretability, since the action associated with geometry closure and the action associated with phase drift remain separable at the diagnostic level. At the same time, the final command remains physically coupled through the common thrust bound applied afterward.

3.6.2 Magnitude Saturation under Limited Thrust

The commanded acceleration must satisfy the available authority constraint

$$\|\mathbf{u}^R\| \le a_{\max}(t_k). \tag{3.39}$$

To enforce feasibility, a norm-based projection is applied:

$$\mathbf{u}^R = \begin{cases} \mathbf{u}^R_{\text{raw}}, & \|\mathbf{u}^R_{\text{raw}}\| \le a_{\max}, \\ a_{\max}\dfrac{\mathbf{u}^R_{\text{raw}}}{\|\mathbf{u}^R_{\text{raw}}\|}, & \text{otherwise.} \end{cases} \tag{3.40}$$

This projection preserves the direction of the requested command while guaranteeing compatibility with the available thrust bound. It is also consistent with the normalization later adopted for imitation-learning actions, which are expressed relative to $a_{\max}$.

3.6.3 Optional Output Smoothing

To reduce chatter introduced by discrete-time execution, an optional first-order smoothing can be applied to the final RTN command:

$$\mathbf{u}^R(t_k) \leftarrow (1-\alpha)\,\mathbf{u}^R(t_{k-1}) + \alpha\,\mathbf{u}^R(t_k), \qquad \alpha \in (0,1]. \tag{3.41}$$

This mechanism is not conceptually essential to the guidance law, but it can improve the regularity of stored expert actions, especially in large dataset-generation campaigns where excessive high-frequency variability is undesirable.

3.6.4 Guardrails and Overrides

A limited set of implementation-level guardrails is added to improve robustness in rare edge cases:

- **enable override**: if guidance is disabled by a higher-level supervisor, impose $\mathbf{u}^R = \mathbf{0}$;
- **near-target hold**: optionally enforce $\mathbf{u}^R = \mathbf{0}$ within a strict handover envelope to avoid unnecessary micro-actuation;

- **altitude-floor protection**: suppress inward radial thrust when approaching a predefined minimum altitude.

These mechanisms are not intended to replace a complete operational fault-protection system. Their role is more limited and consistent with the thesis scope: to prevent rare but undesirable behaviors in long closed-loop simulations used for expert validation and data generation.

3.7 Persistent Handover Detection and Terminal Flag

The expert also evaluates whether far-range rendezvous has progressed sufficiently to justify handover to a subsequent guidance layer, such as a proximity-operations controller. Define

$$\rho(t) = \|\mathbf{r}_c - \mathbf{r}_t\|, \qquad v_{\mathrm{rel}}(t) = \|\mathbf{v}_c - \mathbf{v}_t\|. \tag{3.42}$$

A terminal envelope is introduced through the thresholds

$$\rho \le \rho_{\mathrm{tol}}, \qquad v_{\mathrm{rel}} \le v_{\mathrm{tol}}. \tag{3.43}$$

Because these inequalities may be satisfied only transiently in oscillatory regimes, the expert adopts a persistence criterion: the terminal flag is asserted only if both conditions remain valid over a continuous dwell interval T_{persist}. In practice, an internal timer or counter increments while the terminal conditions are satisfied, resets when either condition is violated, and declares completion once the accumulated dwell exceeds the prescribed persistence time.

This persistence-based logic prevents false positives caused by short-lived alignments and is consistent with the broader emphasis of the thesis on robust, repeatable supervisory behavior.

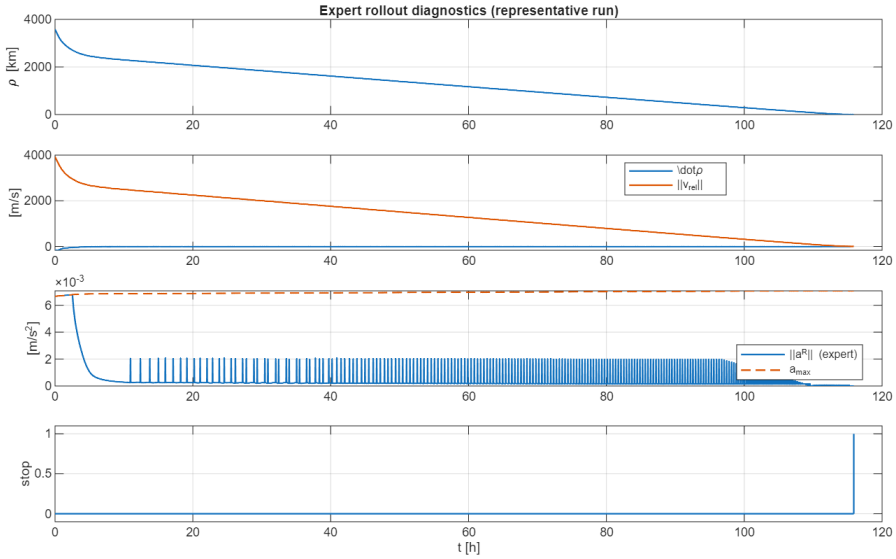

Figure 3.2: Representative expert rollout used to qualitatively validate the closed-loop behavior of the hybrid guidance law. The relative range decreases over time, the commanded acceleration remains bounded by the available authority $a_{\max}(t)$, and the stop flag is asserted only after persistent satisfaction of the terminal conditions.

3.8 Algorithmic Summary

At each update instant t_k, the expert guidance law operates according to the following sequence:

1. read the chaser and target inertial states together with the available acceleration bound $a_{\max}(t_k)$;
2. construct the RTN frame and convert the states into MEE coordinates;
3. compute the geometry error vector $\mathbf{e}_g$ and the scalar indicator e_{geom};
4. evaluate the geometry descent direction $\mathbf{q}_g = -B_{\text{mee}}^{\top} \nabla V_g$ and normalize it when reliable;
5. apply geometry tapering to obtain $\mathbf{u}_{\text{geom}}^{R}$, optionally subject to a predicted-descent consistency check;
6. compute the filtered phase metric $\Delta\lambda_m$ and evaluate phase-readiness logic;
7. if phase is enabled, compute Δa_{cmd}, derive $a_{T,\text{phase}}$, and form $\mathbf{u}_{\text{phase}}^{R}$;

8. sum channel contributions, enforce the norm bound $\|\mathbf{u}^R\| \leq a_{\max}$, and optionally smooth the output;

9. apply guardrails and map the RTN command to inertial coordinates through (3.8);

10. update persistence-based terminal logic and output $\mathbf{a}_c^I(t_k)$ together with the handover flag.

3.9 Tuning Philosophy and Practical Trade-Offs

The expert guidance law contains several tunable parameters: geometry weights W_g, tapering thresholds, phase-gating thresholds, phase time constants, deadbands, saturation caps, numerical safeguards, smoothing factor α, and persistence time T_{persist}. These parameters are not interpreted as arbitrary fitting constants; rather, each group plays a distinct functional role in shaping the balance between convergence speed, smoothness, robustness, and repeatability.

The overall tuning philosophy adopted in this work prioritizes:

1. **robustness and repeatability** over aggressive local convergence;

2. **smooth bounded commands** over high-frequency switching;

3. **physical interpretability** over fully coupled black-box optimization.

This choice is consistent with the intended use of the expert as a demonstration source for imitation learning, where brittle or overly discontinuous commands would degrade dataset quality and complicate surrogate training.

Table 3.1 summarizes the main tuning groups and their qualitative effects.

3.10 Limitations of the Implemented Expert Law

Despite its practical strengths, the proposed expert has several limitations that should be stated explicitly.

3.10.1 Local and Heuristic Nature of the Descent Logic

The geometry channel is based on a local sensitivity-derived descent direction rather than on the solution of a global optimal-control problem. As a result, the law does not guarantee minimum-time or minimum-propellant behavior, and its performance can depend on the selected weights and supervisory thresholds.

Table 3.1: Representative tuning parameters of the expert guidance law and qualitative effects.

Symbol / group	**Role**	**Qualitative effect / trade-off**
$W_g = \mathrm{diag}(w_p, w_f, w_g, w_h, w_k)$	Geometry priorities	Balances size, shape, and plane closure; larger plane weights accelerate cross-track alignment but may delay readiness for phase correction.
$e_{\mathrm{quiet}},\ \gamma$	Geometry tapering	Control the onset and steepness of thrust reduction near convergence; reduce chatter, at the cost of potentially increasing time-to-handover.
$e_{\mathrm{gate}},\ e_{\mathrm{plane}}$	Phase gating	Enable phase correction only when geometry is sufficiently closed; avoid inefficient early phasing, but thresholds that are too strict may delay final timing correction.
λ_{db}	Phase deadband	Suppress phase activity for small residual timing errors; improve regularity, but an excessively large deadband may leave residual along-track mismatch.
$\tau_{\mathrm{phase}},\ \tau_a$	Phase aggressiveness	Set the time scales for drift planning and semi-major-axis servo action; smaller values accelerate phase correction but may increase oscillatory behavior.
$\Delta a_{\max},\ \eta_{a,\max}$	Drift bounds	Limit orbit excursion during phasing; improve regularity and containment, at the expense of reduced phase-correction authority.
$A_{\mathrm{phase,max}}$	Tangential cap	Bounds the tangential contribution of the phase channel; prevents over-actuation but may slow residual phase closure.
α	Output smoothing	Reduces discretization chatter in the final command; excessive smoothing can degrade responsiveness.
T_{persist}	Handover robustness	Prevents false completion declarations due to transient threshold crossings; larger values improve robustness but increase completion latency.
$\varepsilon_q,\ w_{\min}$	Numerical safeguards	Prevent noise-driven thrusting and ill-conditioned sensitivity spikes; overly conservative values may under-actuate in rare configurations.

3.10.2 Approximate Phase Handling

The phase channel relies on first-order relationships among semi-major axis, mean motion, and secular phase drift. This approximation is effective in the low-eccentricity far-range regime considered in this thesis, but it may require retuning or reformulation in strongly perturbed environments or in regimes where the distinction between osculating and mean elements becomes more critical.

3.10.3 Task-Decoupling Assumptions

The hybrid architecture assumes that geometry and phase can be treated as partially decoupled tasks. This is a useful engineering approximation, but not an exact one: geometry and phase remain coupled through the nonlinear orbital dynamics, and the quality of the gating logic directly affects terminal behavior.

3.10.4 No Global Optimality or Formal Stability Proof for the Full Hybrid Supervisor

Although each component of the expert is physically motivated and the geometry channel is locally descent-oriented, the full hybrid supervisor—including filtering, gating, deadbands, smoothing, saturation, and guardrails—is not presented as globally optimal or formally stable. Its value in the present thesis lies instead in providing a robust, interpretable, and computationally reliable expert for simulation-driven learning and benchmarking.

3.11 Why This Expert Is Suitable for Imitation Learning

A central objective of this thesis is not only to design a workable expert, but to design one whose behavior can be meaningfully imitated by a neural surrogate. In this respect, the proposed hybrid guidance law is well suited to imitation learning for several reasons:

- **physical interpretability**: the command can be decomposed into geometry-driven and phase-driven contributions with distinct dynamical roles;
- **bounded feasible outputs**: the final acceleration is explicitly projected onto the admissible set defined by $a_{\max}$;
- **repeatable behavior**: deterministic logic, explicit gating, and numerical safeguards reduce run-to-run variability;

- **meaningful sparsity**: tapering and deadbands generate natural coasting periods instead of noise-driven residual actions;
- **diagnostic transparency**: internal metrics and gate states can be logged and used to interpret both expert failures and surrogate discrepancies.

These properties are valuable not because they make the expert universally optimal, but because they produce supervision targets that are coherent, bounded, and interpretable. This, in turn, improves both the quality of the demonstration dataset and the interpretability of the surrogate-policy errors discussed in the later comparison chapters.

3.12 Chapter Summary

This chapter introduced the expert guidance policy used throughout the thesis and framed it as a Q-law-inspired hybrid strategy for low-thrust far-range rendezvous. The proposed law combines:

1. a weighted geometry-closure channel in MEE space based on a local sensitivity-derived descent direction;
2. a gated phase-correction channel based on tangential drift induced by temporary semi-major-axis offset;
3. feasibility enforcement, optional smoothing and guardrails, and persistence-based handover detection.

The resulting expert is not claimed to be globally optimal, nor formally stable as a complete hybrid supervisor. Its relevance within this thesis lies instead in its robustness, physical interpretability, and suitability for large-scale simulation campaigns and imitation-learning supervision. The next chapter builds on this formulation to define the neural surrogate architecture and the learned policy representation.

Chapter 4

Neural Surrogate Policy Architecture and Inference Wrapper

This chapter presents the neural surrogate policy adopted to approximate the expert guidance law introduced in Chapter 3. The surrogate is designed as a *drop-in command generator*: it receives the same simulator states available to the expert and returns a commanded acceleration to be applied to the chaser dynamics in closed loop.

A central design choice of this thesis is that the surrogate is *not* treated as an end-to-end autonomous controller. Instead, it is embedded within a model-based simulation and guidance environment in which orbital propagation, frame transformations, thrust-to-acceleration scaling, and termination logic remain deterministic and physically structured. The neural network is used exclusively to approximate the expert's *command selection map*. This separation is deliberate: it isolates the learning problem to the policy component that is most difficult to hand-design while preserving the interpretability and reproducibility of the surrounding guidance infrastructure.

In the implemented pipeline:

- the network input is a fixed-dimensional history-stacked feature vector (55×1) built from relative kinematics in the chaser RTN frame;
- the network output is a bounded, dimensionless RTN vector (3×1) generated by a feed-forward MLP with `tanh` output;
- an inference-time wrapper converts the raw network output into a feasible physical acceleration command through deadbanding, magnitude shaping,

confidence-based attenuation, and frame remapping.

Dataset generation and training protocol are described in Chapter 5, while closed-loop evaluation against the expert is reported in Chapter 6. The purpose of the present chapter is narrower and more precise: to define the deployed surrogate policy as an input/output contract, describe the network architecture used to approximate that contract, and document the deterministic post-processing that turns the network output into the command actually applied to the plant.

Policy definition (MLP + deterministic wrapper). Throughout this thesis, the term *neural surrogate policy* refers to the complete deployed module that produces the applied acceleration command. It consists of: (i) a trainable multilayer perceptron (MLP) that predicts a bounded, dimensionless RTN command vector, and (ii) a deterministic inference-time wrapper that enforces feasibility and regulates command magnitude under low-confidence conditions through deadbanding and confidence attenuation (the so-called "airbag"). The wrapper introduces no trainable parameters; however, it is part of the deployed controller and is therefore documented explicitly to ensure fair, reproducible, and physically meaningful closed-loop evaluation.[1]

4.1 Surrogate Role and Interface Contract

The surrogate replaces the expert command generator inside the guidance loop without modifying the surrounding simulation infrastructure. This permits a controlled comparison in which performance differences can be attributed primarily to policy approximation rather than to changes in orbital dynamics, integration logic, or terminal evaluation.

At discrete update instants t_k, the surrogate receives the inertial states of chaser and target,

$$(\mathbf{r}_c^I(t_k),\, \mathbf{v}_c^I(t_k)), \qquad (\mathbf{r}_t^I(t_k),\, \mathbf{v}_t^I(t_k)), \qquad \mu, \tag{4.1}$$

where superscript I denotes inertial coordinates. In addition, the inference wrapper receives the instantaneous available acceleration bound

$$a_{\max}(t_k) = \frac{T}{m(t_k)}, \tag{4.2}$$

together with an enable flag. The surrogate returns a commanded acceleration

$$\mathbf{a}_{\mathrm{NN}}^I(t_k) \in \mathbb{R}^3, \tag{4.3}$$

plus auxiliary diagnostics used for analysis, such as internal command magnitude, thrust fraction, and validity flags.

For reproducibility and safe deployment, the surrogate is defined by a fixed interface contract:

1. **Input construction:** a vector $\mathbf{x}_k \in \mathbb{R}^{55}$ obtained by stacking a $K = 5$ history of instantaneous feature vectors $\boldsymbol{\phi}_k \in \mathbb{R}^{11}$ (Section 4.2);

2. **Preprocessing:** frozen standardization vectors $(\boldsymbol{\mu}_x, \boldsymbol{\sigma}_x)$ estimated from the training set, followed by component-wise clipping to $[-8,8]$ (Section 4.3);

3. **Network output semantics:** a bounded vector $\mathbf{y}_k \in [-1,1]^3$ representing a dimensionless RTN command proposal (Sections 4.4 and 4.5);

4. **Post-processing:** deterministic deadbanding, confidence-based magnitude attenuation, and scaling by $a_{\max}(t_k)$ to obtain the applied RTN acceleration (Section 4.5);

5. **Frame mapping:** RTN-to-inertial conversion through the chaser RTN frame $C_{I \leftarrow R}(t_k)$ (Section 4.2).

Any inconsistency in feature ordering, normalization vectors, or frame conventions between training and deployment is typically catastrophic. For this reason, these elements are treated not as ancillary implementation details but as part of the surrogate-policy definition itself.

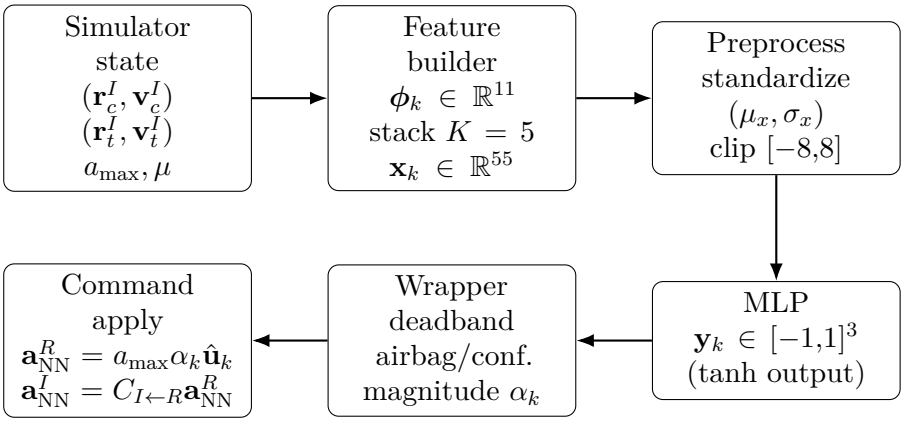

Figure 4.1: Inference pipeline of the deployed neural surrogate. The policy is defined as an MLP combined with a deterministic wrapper: the MLP proposes a bounded RTN command vector, while the wrapper regulates thrust magnitude under low-confidence conditions, enforces feasibility, and maps the result to inertial acceleration.

4.2 Input Representation: History-Stacked RTN Relative Features

4.2.1 RTN Frame Definition

All surrogate inputs are constructed in the local orbital RTN frame of the chaser. This choice preserves physical interpretability, aligns the input representation with the command semantics used by the expert, and reduces the burden on the network to discover a meaningful directional decomposition from inertial coordinates alone.

At time t_k, define the chaser RTN basis as

$$\hat{\mathbf{R}}_k = \frac{\mathbf{r}_c^I(t_k)}{\|\mathbf{r}_c^I(t_k)\|}, \tag{4.4}$$

$$\hat{\mathbf{N}}_k = \frac{\mathbf{r}_c^I(t_k) \times \mathbf{v}_c^I(t_k)}{\|\mathbf{r}_c^I(t_k) \times \mathbf{v}_c^I(t_k)\|}, \tag{4.5}$$

$$\hat{\mathbf{T}}_k = \hat{\mathbf{N}}_k \times \hat{\mathbf{R}}_k, \tag{4.6}$$

with inertial-to-RTN rotation

$$C_{R\leftarrow I}(t_k) = \begin{bmatrix} \hat{\mathbf{R}}_k^\top \\ \hat{\mathbf{T}}_k^\top \\ \hat{\mathbf{N}}_k^\top \end{bmatrix}. \tag{4.7}$$

The inverse mapping used at deployment is

$$C_{I\leftarrow R}(t_k) = \begin{bmatrix} \hat{\mathbf{R}}_k & \hat{\mathbf{T}}_k & \hat{\mathbf{N}}_k \end{bmatrix}. \tag{4.8}$$

4.2.2 Instantaneous Feature Vector $\phi_k \in \mathbb{R}^{11}$

Let the inertial relative position and velocity be

$$\Delta\mathbf{r}_k^I = \mathbf{r}_t^I(t_k) - \mathbf{r}_c^I(t_k), \qquad \Delta\mathbf{v}_k^I = \mathbf{v}_t^I(t_k) - \mathbf{v}_c^I(t_k). \tag{4.9}$$

Their RTN components are computed as

$$\Delta\mathbf{r}_k^R = C_{R\leftarrow I}(t_k)\,\Delta\mathbf{r}_k^I, \qquad \Delta\mathbf{v}_k^R = C_{R\leftarrow I}(t_k)\,\Delta\mathbf{v}_k^I. \tag{4.10}$$

Define the scalar relative range and a closing-rate-like quantity:

$$\rho_k \triangleq \|\Delta\mathbf{r}_k^R\|, \tag{4.11}$$

$$\dot{\rho}_k \triangleq \begin{cases} \dfrac{(\Delta\mathbf{r}_k^R)^\top \Delta\mathbf{v}_k^R}{\rho_k}, & \rho_k > \varepsilon_\rho, \\ 0, & \rho_k \le \varepsilon_\rho, \end{cases} \tag{4.12}$$

where ε_ρ is a small numerical safeguard.

In addition, define three chaser orbital-scale quantities,

$$r_k \triangleq \|\mathbf{r}_c^I(t_k)\|, \qquad v_k \triangleq \|\mathbf{v}_c^I(t_k)\|, \qquad n_k \triangleq \sqrt{\frac{\mu}{r_k^3}}, \tag{4.13}$$

where n_k is used as a compact mean-motion-like scale.

The instantaneous feature vector is then

$$\boldsymbol{\phi}_k \triangleq \begin{bmatrix} (\Delta\mathbf{r}_k^R)^\top & (\Delta\mathbf{v}_k^R)^\top & \rho_k & \dot{\rho}_k & r_k & v_k & n_k \end{bmatrix}^\top \in \mathbb{R}^{11}. \tag{4.14}$$

The rationale behind this feature set is pragmatic. It includes the relative translational state in physically meaningful RTN coordinates, together with a small number of scalar orbital-scale quantities that help disambiguate the dynamical regime. The resulting representation is compact enough for efficient feed-forward inference, yet structured enough to preserve the main information required for command prediction.

4.2.3 History Stacking and Warm-Up Validity

Rather than using a recurrent architecture, the surrogate provides short-term temporal context to a feed-forward network through a fixed-length history stack. In the implemented model,

$$K = 5, \qquad d_\phi = 11, \qquad d_x = K\, d_\phi = 55. \tag{4.15}$$

The stacked unnormalized input is defined as

$$\mathbf{x}_k \triangleq \begin{bmatrix} \boldsymbol{\phi}_k^\top & \boldsymbol{\phi}_{k-1}^\top & \cdots & \boldsymbol{\phi}_{k-K+1}^\top \end{bmatrix}^\top \in \mathbb{R}^{55}, \tag{4.16}$$

using a *most-recent-first* ordering. In the implementation, this vector is maintained through a persistent shift-register buffer: at each update, older entries are shifted and the newest feature vector $\boldsymbol{\phi}_k$ is inserted at the top.

Because the buffer is initially empty, the surrogate exposes a validity flag,

$$\mathrm{valid}_{\mathrm{NN}}(k) = \begin{cases} 1, & k \geq K \text{ and guidance enabled}, \\ 0, & \text{otherwise}. \end{cases} \tag{4.17}$$

which indicates whether a full input history is available. This flag is later used by the selector logic in Section 4.6 to prevent network activation before the temporal window has been fully populated. In practice, a warm-up horizon of $K-1$ steps is sufficient.

Table 4.1: Surrogate input dimensionality. The policy input is a fixed-size history stack that provides short-term temporal context while preserving feed-forward inference and simple deployment.

Quantity	**Value**
Instantaneous feature dimension d_ϕ	11
History length K	5
Network input dimension $d_x = K\, d_\phi$	55

4.3 Input Normalization and Clipping

The stacked input $\mathbf{x}_k$ combines heterogeneous physical quantities, including position, velocity, range, and orbital-scale variables. Feeding these values directly into an MLP would result in poor conditioning and unstable optimization. The surrogate therefore applies element-wise standardization:

$$\mathbf{x}_k^{\text{norm}}(i) = \frac{\mathbf{x}_k(i) - \mu_x(i)}{\sigma_x(i)}, \qquad i = 1, \ldots, 55, \tag{4.18}$$

where $(\boldsymbol{\mu}_x, \boldsymbol{\sigma}_x)$ are computed from the training dataset and then frozen for deployment.

To prevent division by very small or non-finite values, the standard deviation is safeguarded according to

$$\sigma_x(i) \leftarrow \max\Big(\sigma_x(i), \sigma_{\min}\Big), \tag{4.19}$$

with $\sigma_{\min} > 0$.

After standardization, the input is clipped component-wise:

$$\mathbf{x}_k^{\text{clip}} = \text{clip}(\mathbf{x}_k^{\text{norm}},\, -x_{\max},\, x_{\max})\,, \qquad x_{\max} = 8, \tag{4.20}$$

and $\mathbf{x}_k^{\text{clip}}$ is the actual vector fed to the network.

Clipping rationale. The clipping threshold $x_{\max} = 8$ is chosen as a conservative bound: it is large enough to avoid distorting the bulk of the training distribution, yet finite enough to prevent rare outliers from producing extreme internal activations. In closed-loop operation, repeated activation of this clipping bound also serves as a practical indicator of distribution shift, a point that becomes relevant when defining the confidence-based attenuation mechanism introduced later in this chapter.

4.4 Network Architecture

The surrogate policy is implemented as a feed-forward multilayer perceptron (MLP) that maps the clipped input vector $\mathbf{x}_k^{\text{clip}} \in \mathbb{R}^{55}$ to a three-dimensional RTN output:

$$\mathbf{y}_k = \pi_\theta(\mathbf{x}_k^{\text{clip}}) \in \mathbb{R}^3. \tag{4.21}$$

The final model used in the closed-loop experiments adopts the following architecture:

- input dimension 55, with no internal normalization layer since normalization is handled explicitly by (4.18)–(4.20);
- two fully connected hidden layers of width 256;
- layer normalization after each hidden affine transformation;
- ReLU activation after each layer normalization;
- a final fully connected layer to 3 outputs followed by `tanh`.

[13, 14]

In compact notation,

$$\mathbf{z}_1 = \mathrm{ReLU}\Big(\mathrm{LN}\Big(W_1\mathbf{x}_k^{\text{clip}} + \mathbf{b}_1\Big)\Big)\,, \tag{4.22}$$

$$\mathbf{z}_2 = \mathrm{ReLU}(\mathrm{LN}(W_2\mathbf{z}_1 + \mathbf{b}_2))\,, \tag{4.23}$$

$$\mathbf{y}_k = \tanh(W_3\mathbf{z}_2 + \mathbf{b}_3)\,, \tag{4.24}$$

with $\mathbf{y}_k \in [-1{,}1]^3$ component-wise.

The choice of a feed-forward MLP, rather than a recurrent or transformer-based architecture, reflects the constraints of the deployment setting. The model must remain lightweight, fast to evaluate, straightforward to embed in Simulink, and fully compatible with deterministic preprocessing and code-generation-oriented deployment. The history stack described in Section 4.2.3 provides limited temporal context without sacrificing this simplicity.

Simulink deployment. The trained network is executed through a Deep Learning `Predict` block configured to load the model from a MAT-file. The surrounding logic-feature construction, normalization, clipping, wrapper operations, and command remapping are implemented in code-generation-compatible MATLAB Function blocks.

4.5 Output Semantics and Inference-Time Post-Processing

The raw network output $\mathbf{y}_k$ is interpreted as a *dimensionless RTN command proposal*. It is not directly applied as physical acceleration. Instead, the final command is produced by a deterministic wrapper that enforces feasibility, suppresses residual micro-thrusting, and reduces thrust authority when the input appears poorly represented by the training distribution.

A critical point is that component-wise boundedness, i.e. $\mathbf{y}_k \in [-1,1]^3$, does not imply $\|\mathbf{y}_k\| \leq 1$. The deployed surrogate therefore separates *direction* and *magnitude*: the network proposes an RTN direction, while the wrapper determines how much of the currently available thrust authority should be used.

4.5.1 Component-Wise Safety Clamp

Although the network already employs a `tanh` output layer, the implementation applies an explicit component-wise clamp,

$$y_k(i) \leftarrow \mathrm{clip}(y_k(i), -1,1), \qquad i \in \{1,2,3\}, \tag{4.25}$$

to preserve boundedness in the presence of numerical anomalies and to simplify deterministic deployment.

4.5.2 Magnitude Shaping with Deadband

Define the raw output norm

$$u_k \triangleq \|\mathbf{y}_k\|. \tag{4.26}$$

Very small outputs are suppressed by a deadband:

$$\text{if } u_k < u_{\text{dead}} \text{ then } \mathbf{a}_{\text{NN}} = \mathbf{0}, \qquad u_{\text{dead}} = 0.05. \tag{4.27}$$

This reduces chatter and prevents the plant from receiving persistent micro-thrusting commands near zero.

For $u_k \geq u_{\text{dead}}$, the commanded thrust fraction is defined as

$$\alpha_k \triangleq \mathrm{clip}\left(\frac{u_k - u_{\text{dead}}}{1 - u_{\text{dead}}},\, 0,\, 1\right), \tag{4.28}$$

and the RTN direction is

$$\hat{\mathbf{u}}_k \triangleq \frac{\mathbf{y}_k}{u_k + \varepsilon_u}, \tag{4.29}$$

with small $\varepsilon_u > 0$.

This construction assigns the network two distinct roles: the orientation of $\mathbf{y}_k$ determines the commanded direction, while its norm determines a bounded thrust fraction after deadbanding and scaling.

4.5.3 Out-of-Distribution "Airbag" Attenuation

Because the normalized input is clipped at ± 8, strong activation of the clipping boundary indicates that the current state may lie outside the bulk of the training distribution. This condition is used as a conservative proxy for reduced confidence.

Define the saturation ratio

$$s_k \triangleq \frac{\max_i |x_k^{\text{clip}}(i)|}{x_{\max}} \in [0,1], \qquad x_{\max} = 8. \tag{4.30}$$

A confidence factor is then computed as

$$c_k \triangleq \text{clip}(1 - 0.8\, s_k,\, c_{\min},\, 1)\,, \qquad c_{\min} = 0.30, \tag{4.31}$$

and applied to the thrust fraction:

$$\alpha_k \leftarrow \text{clip}(\alpha_k\, c_k,\, 0,\, 1). \tag{4.32}$$

Intuitively, when the surrogate operates near the clipping boundary ($s_k \approx 1$), the applied thrust fraction is attenuated, down to a minimum of 30% of its unattenuated value. This mechanism does not certify correctness; rather, it acts as a conservative safeguard that reduces the aggressiveness of the surrogate in regions where the learned mapping is more likely to be extrapolating.

4.5.4 RTN Physical Command and Inertial Mapping

Given the instantaneous acceleration bound $a_{\max}(t_k)$, the applied RTN acceleration is

$$\mathbf{a}_{\text{NN}}^{R}(t_k) = a_{\max}(t_k)\, \alpha_k\, \hat{\mathbf{u}}_k, \tag{4.33}$$

which guarantees $\|\mathbf{a}_{\text{NN}}^{R}(t_k)\| \le a_{\max}(t_k)$ by construction. The inertial command is then obtained as

$$\mathbf{a}_{\text{NN}}^{I}(t_k) = C_{I \leftarrow R}(t_k)\, \mathbf{a}_{\text{NN}}^{R}(t_k). \tag{4.34}$$

If guidance is disabled, or if $a_{\max}(t_k)$ is non-finite or non-positive, the wrapper returns

$$\mathbf{a}_{\text{NN}}^{I}(t_k) = \mathbf{0}. \tag{4.35}$$

This final mapping completes the definition of the deployed policy: the MLP predicts a bounded RTN command proposal, while the wrapper converts that proposal into a feasible inertial acceleration command consistent with the thrust authority available at each update.

4.6 Closed-Loop Integration and Expert/NN Selection

For controlled evaluation and debugging, the simulation computes both the expert command $\mathbf{a}_E^I(t_k)$ and the neural command $\mathbf{a}_{\mathrm{NN}}^I(t_k)$ in parallel. A deterministic selector then determines which one is actually applied to the plant.

Let $\mathrm{valid}_{\mathrm{NN}}(k)$ denote the history-buffer validity flag, and let `useNN_for_plant` be a run-time switch. The applied acceleration is

$$\mathbf{a}_{\mathrm{cmd}}^I(t_k) = \begin{cases} \mathbf{a}_{\mathrm{NN}}^I(t_k), & \text{if } \texttt{useNN\_for\_plant} = 1,\ \mathrm{valid}_{\mathrm{NN}}(k) = 1,\ k > k_{\mathrm{warm}}, \\ \mathbf{a}_E^I(t_k), & \text{otherwise,} \end{cases} \tag{4.36}$$

where k_{warm} is a warm-up threshold, typically chosen as $K - 1$.

Warm-up validity due to history stacking. Because the network input stacks a $K = 5$ history window, the input is not valid during the first $K - 1$ guidance updates after activation. A deterministic warm-up phase is therefore enforced: until the history buffer is fully populated, the plant is driven by the expert or another safe fallback. Only after this transient does the NN command become eligible for application. This avoids introducing an artificial activation discontinuity unrelated to the learned mapping itself.

4.7 Implementation Notes and Practical Considerations

4.7.1 Code-Generation Compatibility

All MATLAB Function blocks involved in feature construction and wrapper post-processing are written to remain compatible with code generation (`#codegen`). This includes explicit shape handling, guarded norms, and simple deterministic control flow.

4.7.2 Logging and Diagnostics

The surrogate pipeline exposes internal diagnostics used for debugging and for the interpretation of closed-loop behavior. In particular, the implementation logs:

- statistics of $\mathbf{x}_k^{\mathrm{norm}}$ and $\mathbf{x}_k^{\mathrm{clip}}$, useful to monitor clipping activity and possible distribution shift;
- raw output norm u_k and thrust fraction α_k;

- the validity flag $\text{valid}_{\text{NN}}(k)$;
- expert and NN command traces, enabling direction and magnitude comparisons over time.

These signals are practically important because many closed-loop failures do not originate from the network weights alone. In deployment, preprocessing mismatch, invalid scaler vectors, history-stack inconsistencies, or persistent clipping can degrade performance even when the trained MLP is otherwise well behaved.

4.8 Limitations and Expected Failure Modes

The surrogate policy inherits both approximation error and integration sensitivity. The most relevant expected failure modes for the implemented architecture are the following:

- **distribution shift**: states outside the effective support of the dataset can produce strong input clipping, which triggers airbag attenuation and may slow or distort convergence;
- **long-horizon accumulation**: small systematic directional biases may compound over long rendezvous durations, especially through tangential acceleration effects;
- **dependence on preprocessing correctness**: feature ordering, scaler vectors, and history stacking must match exactly between training and deployment; mismatches are typically catastrophic;
- **warm-up sensitivity**: early transient behavior depends on history-buffer fill and selector logic, so improper warm-up handling can introduce discontinuities at surrogate activation.

These limitations motivate the evaluation methodology adopted in Chapter 6. The surrogate is not assessed solely through one-step prediction error, but through closed-loop trajectory behavior, terminal success statistics, and diagnostic indicators that reveal whether the deployed policy remains within the regime represented by the training data.[1]

4.9 Chapter Summary

This chapter defined the neural surrogate policy as a precise deployed module composed of:

1. a history-stacked RTN feature builder producing $\mathbf{x}_k \in \mathbb{R}^{55}$ from chaser and target inertial states;
2. an explicit normalization and clipping pipeline producing $\mathbf{x}_k^{\text{clip}}$;
3. a feed-forward MLP mapping $\mathbf{x}_k^{\text{clip}}$ to a bounded RTN output $\mathbf{y}_k \in [-1,1]^3$;
4. a deterministic inference-time wrapper converting $\mathbf{y}_k$ into a feasible inertial acceleration command through deadbanding, magnitude shaping, confidence attenuation, and RTN-to-inertial remapping;
5. a deterministic selector enabling controlled switching between expert and surrogate during closed-loop simulations.

The main conceptual point is that the deployed controller is not the MLP alone, but the full surrogate module obtained by combining the MLP with deterministic wrapper logic. The next chapter builds on this deployed contract to describe how the dataset artifacts, frozen scaler statistics, and deterministic data splits are generated and curated consistently with the surrogate interface.

Chapter 5

Dataset Generation

This chapter describes the dataset-generation pipeline used to train the neural surrogate policy introduced in Chapter 4. The dataset is obtained by rolling out the expert guidance law of Chapter 3 in the Simulink closed-loop environment, logging state–action trajectories, and converting them into a chunked representation suitable for reproducible training, validation, and regression testing.

For long-horizon low-thrust guidance, dataset construction is not a secondary implementation step but a methodological component of the learning framework. Small inconsistencies between offline dataset generation and online closed-loop deployment can invalidate the learned policy even when the network architecture and training procedure are otherwise correct. For this reason, the pipeline is designed around two main principles:

- **semantic consistency**: feature construction, frame conventions, normalization, and label definitions must be identical between offline data generation and online inference;

- **traceable artifact management**: dataset chunks, scaler statistics, and train/validation/test partitions must be stored in a deterministic and reproducible form, enabling controlled retraining and reliable comparison across experiments.

The present chapter therefore addresses not only how samples are generated, but also how they are filtered, normalized, partitioned, and validated before training. This is particularly important in the present application, where long-horizon closed-loop behavior is sensitive to small directional errors that may be invisible in one-step prediction metrics alone.

5.1 Dataset Objective

The dataset is designed to approximate the expert command-generation map with a neural surrogate operating inside the same guidance loop. Each supervised sample has the form

$$(\mathbf{x}_k,\, \mathbf{y}_k), \qquad \mathbf{x}_k \in \mathbb{R}^{55}, \quad \mathbf{y}_k \in \mathbb{R}^3, \tag{5.1}$$

where:

- $\mathbf{x}_k$ is a fixed-dimensional, history-stacked feature vector constructed from relative kinematics in the chaser RTN frame, as defined in Chapter 4;
- $\mathbf{y}_k$ is the expert commanded acceleration expressed in RTN coordinates and normalized by the instantaneous acceleration authority $a_{\max}(t) = T/m(t)$.

This formulation is chosen so that the supervised learning problem matches the deployed surrogate-policy contract as closely as possible. In other words, the network is trained to reproduce the same type of command that it will later be required to generate online, under the same frame conventions and normalization rules.

The resulting dataset supports two complementary types of evaluation:

- **open-loop evaluation**, in which the surrogate is compared to the expert on expert-generated trajectories and therefore on approximately the same state distribution;
- **closed-loop evaluation**, in which the surrogate is inserted online and induces its own visited-state distribution, thereby exposing the effects of approximation error, covariate shift, and error compounding over sequential decision steps.[15, 1]

This distinction is central to the thesis. Open-loop metrics quantify how well the learned mapping reproduces the expert on demonstrated states, whereas closed-loop metrics determine whether this approximation remains operationally meaningful when rolled out over long guidance horizons. This separation is standard in imitation learning, where a policy that performs well on expert-generated samples may still fail after deployment because it progressively visits states that were weakly represented in the original supervision dataset.[15, 1]

5.2 Statistical Planning: Hoeffding–Chernoff Bounds for Monte Carlo Sizing

A practical question in dataset generation is the number of rollouts required to estimate performance metrics with meaningful confidence. Consider a Bernoulli

success variable $S \in \{0,1\}$ defined at the rollout level, where $S = 1$ if the expert or surrogate reaches the far-range handover condition within the simulation horizon. Let $p = \mathbb{P}(S = 1)$ be the true success probability and let

$$\hat{p} = \frac{1}{N} \sum_{i=1}^{N} S_i \tag{5.2}$$

be its empirical estimate over N independent rollouts.

For bounded random variables, a Hoeffding-type inequality gives [16]

$$\mathbb{P}\left(|\hat{p} - p| \geq \varepsilon\right) \leq 2\exp(-2N\varepsilon^2). \tag{5.3}$$

Therefore, to guarantee

$$|\hat{p} - p| \leq \varepsilon \tag{5.4}$$

with confidence at least $1 - \delta$, it is sufficient to choose

$$N \geq \frac{1}{2\varepsilon^2} \ln\left(\frac{2}{\delta}\right). \tag{5.5}$$

This bound is not used as a sharp optimal design rule, but as a practical sizing guideline for Monte Carlo campaigns. In particular, it helps determine how many rollouts are required to:

- estimate expert and surrogate success rates with controlled uncertainty;
- detect performance degradation after retraining or dataset aggregation;
- compare alternative dataset variants without relying on anecdotal single-run evidence.

Similar reasoning can be applied to other bounded rollout-level metrics, such as normalized thrust-on fraction or feature-saturation ratio, after appropriate normalization to [0,1]. The key methodological point is that rollout count should be justified quantitatively rather than chosen arbitrarily, especially when differences between surrogate variants may be subtle at the one-step level but operationally significant in closed loop.

Practical sizing adopted in this work. In this thesis, the Hoeffding-style bound of (5.5) was not used as a strict optimal-design rule, but as a practical guideline to select the order of magnitude of the Monte Carlo campaign. Based on this rationale, the comparative rollout campaigns were sized to

$$N = 1000 \tag{5.6}$$

independent simulations.

This choice was considered a reasonable compromise between statistical resolution and computational cost. For example, by rearranging (5.5), a campaign of $N = 1000$ rollouts implies that the empirical success rate $\hat{p}$ estimates the true success probability p with an error on the order of a few percentage points. In particular, the Hoeffding bound gives

$$\varepsilon_{95\%} \approx \sqrt{\frac{1}{2N} \ln\left(\frac{2}{0.05}\right)} \approx 0.043, \tag{5.7}$$

and

$$\varepsilon_{99\%} \approx \sqrt{\frac{1}{2N} \ln\left(\frac{2}{0.01}\right)} \approx 0.052. \tag{5.8}$$

Therefore, with 1000 runs, rollout-level success rates can be interpreted with a conservative uncertainty of roughly ± 4–5 percentage points, depending on the desired confidence level.

For the purposes of this thesis, this accuracy was judged sufficient to distinguish practically meaningful differences between expert and surrogate closed-loop behavior, while keeping simulation time compatible with iterative dataset generation, retraining, and regression testing. The same order of magnitude was also considered adequate for bounded secondary metrics evaluated at rollout level, provided that they were normalized consistently.

5.3 Simulation Logging and Raw Signal Extraction

Dataset samples are extracted from Simulink closed-loop rollouts by logging a compact set of input and output signals. The logging interface is intentionally kept simple and robust so that data extraction remains stable across multiple simulation batches and code versions.

Two main logs are stored:

- an **inputs log**, containing chaser and target inertial states together with auxiliary physical scalars;
- an **outputs log**, containing the expert commanded acceleration and the terminal stop flag.

A dedicated unpacking routine, implemented in `unpack_io_vectors.m`, reconstructs the corresponding sample-wise arrays and enforces consistent signal dimensions.

Inputs. The inputs log contains at least the following columns:

$$\left[\begin{array}{cccccc} \mathbf{r}_c^I & \mathbf{v}_c^I & \mathbf{r}_t^I & \mathbf{v}_t^I & a_{\max} & t_{JD} \quad m(t) \end{array}\right], \tag{5.9}$$

namely: inertial position and velocity of chaser and target, instantaneous acceleration authority $a_{\max}(t)$, Julian-date tag used by the expert, and current mass.

Outputs. The outputs log contains at least:

$$\left[\begin{array}{cc} \mathbf{a}_{\text{exp}}^I & \text{stop} \end{array}\right], \tag{5.10}$$

where $\mathbf{a}_{\text{exp}}^I$ is the expert inertial acceleration command and stop $\in \{0,1\}$ denotes the persistence-based terminal flag.

Time alignment. The unpacking stage enforces a common sample length across inputs, outputs, and time tags. This prevents silent time-index mismatches that would otherwise corrupt feature/label pairing and compromise subsequent training. In a long-horizon guidance dataset, even a one-step misalignment between states and commands is effectively catastrophic, because it changes the meaning of the supervised mapping being learned.

5.4 Feature Construction and History Stacking

To preserve semantic consistency between offline learning and online deployment, the dataset uses exactly the same feature construction adopted by the deployed surrogate in Chapter 4. This requirement is fundamental: if the offline feature definition differs from the online inference contract, training and deployment effectively target different functions.

At each sample time t_k, the chaser RTN frame is built from $(\mathbf{r}_c^I, \mathbf{v}_c^I)$, and the relative inertial vectors are transformed as

$$\Delta\mathbf{r}_k^R = C_{R\leftarrow I}(t_k)\left(\mathbf{r}_t^I - \mathbf{r}_c^I\right), \qquad \Delta\mathbf{v}_k^R = C_{R\leftarrow I}(t_k)\left(\mathbf{v}_t^I - \mathbf{v}_c^I\right). \tag{5.11}$$

The instantaneous feature vector $\boldsymbol{\phi}_k \in \mathbb{R}^{11}$ is then defined as

$$\boldsymbol{\phi}_k \triangleq \left[(\Delta\mathbf{r}_k^R)^\top \quad (\Delta\mathbf{v}_k^R)^\top \quad \rho_k \quad \dot{\rho}_k \quad r_k \quad v_k \quad n_k\right]^\top, \tag{5.12}$$

where

$$\rho_k = \|\Delta\mathbf{r}_k^R\|, \qquad \dot{\rho}_k = \frac{(\Delta\mathbf{r}_k^R)^\top \Delta\mathbf{v}_k^R}{\max(\rho_k, \varepsilon_\rho)}, \tag{5.13}$$

and

$$r_k = \|\mathbf{r}_c^I\|, \qquad v_k = \|\mathbf{v}_c^I\|, \qquad n_k = \sqrt{\frac{\mu}{r_k^3}}. \tag{5.14}$$

As discussed in Chapter 4, the instantaneous feature vector is not fed to the network directly. Instead, a fixed history window of length $K = 5$ is stacked to provide short-term temporal context:

$$\mathbf{x}_k \triangleq \begin{bmatrix} \boldsymbol{\phi}_k^\top & \boldsymbol{\phi}_{k-1}^\top & \cdots & \boldsymbol{\phi}_{k-K+1}^\top \end{bmatrix}^\top \in \mathbb{R}^{55}. \tag{5.15}$$

Samples collected before the history buffer is fully populated correspond to a warm-up phase and are either excluded or explicitly flagged as invalid. This mirrors the deployment logic of the surrogate and ensures that training samples are drawn from the same valid input regime used during online inference.

5.5 Label Definition and Normalization

The expert command is logged in inertial coordinates, whereas the neural surrogate is trained to predict a normalized RTN command. The label-construction stage therefore performs two operations: frame conversion and normalization by the instantaneous available acceleration.

First, the expert inertial command is mapped to RTN:

$$\mathbf{a}_{\text{exp}}^R(t_k) = C_{R\leftarrow I}(t_k)\, \mathbf{a}_{\text{exp}}^I(t_k). \tag{5.16}$$

Then the supervised label is defined as

$$\mathbf{y}_k = \frac{\mathbf{a}_{\text{exp}}^R(t_k)}{\bar{a}_{\max}(t_k)}, \tag{5.17}$$

where the safeguarded acceleration bound

$$\bar{a}_{\max}(t_k) = \max(a_{\max}(t_k), \varepsilon_a) \tag{5.18}$$

prevents division by degenerate values.

This normalization is important for two reasons. First, it makes the target largely independent of slow variations in vehicle mass. Second, it aligns the label semantics with the deployed surrogate, whose output is interpreted as a dimensionless RTN command later rescaled by the current thrust authority.

Thrust-on mask. For several stages of training and analysis, it is useful to distinguish active-thrust samples from coasting phases. A thrust-on indicator is defined as

$$\text{thr\_on}(k) \triangleq (\|\mathbf{y}_k\| > \varepsilon_{\text{thr}}), \tag{5.19}$$

with small threshold ε_{thr}.

Direction-only vs. direction+magnitude targets. The stored labels $\mathbf{y}_k$ encode both command direction and magnitude fraction. However, some training configurations emphasize direction more explicitly by mapping thrust-on targets to unit vectors:

$$\mathbf{y}_{k,\mathrm{dir}} = \begin{cases} \mathbf{y}_k/\|\mathbf{y}_k\|, & \mathrm{thr\_on}(k) = 1, \\ \mathbf{0}, & \mathrm{thr\_on}(k) = 0, \end{cases} \tag{5.20}$$

and then optimizing a cosine-based loss. This option is motivated by the closed-loop behavior observed later in the thesis: over long horizons, directional bias can be more destabilizing than moderate magnitude mismatch, especially in the tangential channel where small persistent errors accumulate over time.

5.6 Choice of Dataset Parameters and Variation Strategy

A dataset built from a single nominal trajectory would be insufficient for the present problem. The surrogate must instead learn a command map that remains meaningful across a range of initial geometries and convergence conditions. For this reason, the dataset is generated from multiple independent rollouts under dispersed initial conditions and parameter variations.

The variation strategy is designed to cover:

- different initial relative geometry and phasing conditions;
- different difficulty levels, including mixed plane, shape, and along-track mismatch;
- variability in thrust-to-mass ratio through the time evolution of $m(t)$ and therefore of $a_{\max}(t)$.

Each rollout is assigned a simulation identifier and treated as a coherent trajectory-level unit. This is important because neighboring samples within a rollout are strongly correlated; therefore, dataset partitioning should ideally be performed at the rollout level rather than at the individual-sample level. Such a design reduces information leakage between training and validation subsets and yields a more honest estimate of generalization.

When chunking is performed through file-level partitioning, rollout order is randomized using a fixed random seed so that dataset composition remains repeatable across retraining runs.

Final campaign adopted in this thesis. In the final dataset campaign used in this work, the variation strategy was instantiated through a Monte Carlo set of $N = 1000$ independent expert rollouts. The initial conditions were dispersed around the nominal far-range rendezvous scenario so as to cover different combinations of relative geometry mismatch and phase offset, including both near-nominal cases and more demanding mixed-correction trajectories. Additional variability was naturally introduced through the evolution of the spacecraft mass and therefore of the available acceleration bound $a_{\max}(t)$. The objective of this campaign was not to sample the orbital state space uniformly, which would be impractical, but to populate the subset of states most relevant to the expert-guided rendezvous problem considered in this thesis.

5.7 Artifact and Variable Management: Chunked Storage, Frozen Scalers, and Deterministic Splits

Large-scale simulation campaigns can easily produce millions of samples, making a monolithic dataset impractical to store, inspect, and update. The dataset is therefore stored in a *chunked format*, where each chunk contains a subset of samples extracted from one or more rollouts:

$$\texttt{chunk\_\#ID.mat}: \quad \{X_c,\ Y_c\}. \tag{5.21}$$

Here X_c contains stacked inputs and Y_c contains corresponding labels, with robust shape handling during training to accommodate either row-major or column-major storage.

Chunking serves several purposes:

- it avoids costly monolithic merges;
- it simplifies incremental dataset extension, for instance during DAgger-style aggregation [1];
- it enables traceable bookkeeping at the file level;
- it supports deterministic train/validation/test splits without requiring full in-memory reconstruction of the dataset.

Rollout-level partitioning. In this work, train/validation/test partitioning is performed at the rollout level rather than at the individual-sample level. This prevents temporally adjacent samples from the same trajectory from leaking across subsets and therefore yields a more realistic estimate of generalization performance.

Scaler artifact. The input scaler is stored separately as a single authoritative artifact containing

$$\boldsymbol{\mu}_x \in \mathbb{R}^{55}, \qquad \boldsymbol{\sigma}_x \in \mathbb{R}^{55}, \tag{5.22}$$

denoted in the implementation as `muXv` and `sdXv`. These statistics are computed using the training subset only and are then frozen for all validation, testing, and closed-loop inference.

This distinction is methodologically important. The scaler is not merely a convenience for optimization; it is part of the deployed policy contract. Recomputing it on different subsets would alter the effective meaning of the network input and break comparability across experiments.

5.8 Filtering, Balancing, and Robustification

Before training, the raw logged samples are filtered and processed to remove invalid data and reduce undesirable imbalance effects.

The main quality-control operations include:

- removal of non-finite samples (NaN/Inf) in features or labels;
- rejection or masking of warm-up samples collected before history completion;
- optional rejection of samples associated with clearly unphysical simulator states or failed propagation;
- optional removal of post-stop samples, depending on the intended training configuration.

Input normalization and clipping. Each feature vector is standardized and clipped according to the same rule used at deployment:

$$\mathbf{x}_{\text{norm}}(i) = \frac{\mathbf{x}(i) - \mu_x(i)}{\max(\sigma_x(i), \sigma_{\min})}, \qquad \mathbf{x}_{\text{clip}} = \text{clip}(\mathbf{x}_{\text{norm}}, -8, +8). \tag{5.23}$$

This step is important not only for optimization stability, but also for semantic alignment with the inference wrapper of Chapter 4. In effect, the network is trained on the same clipped input domain that it will encounter online.

Thrust-on emphasis. Long-horizon rendezvous trajectories often contain extended coasting phases. If all samples are weighted uniformly, such phases can dominate the dataset and bias the learning problem toward predicting near-zero outputs. A practical mitigation is to emphasize thrust-on samples in the loss function, for example by assigning larger weights when $\text{thr\_on}(k) = 1$. This does not change the dataset itself, but it changes how the dataset is used during optimization and helps preserve directional fidelity in the command-active regime.

Robustification rationale. These filtering and balancing steps should not be interpreted as arbitrary cosmetic cleanup. In the present problem, the surrogate is highly sensitive to rare semantic inconsistencies and to imbalance between command-active and coast-dominated regimes. Robustification therefore improves not only numerical conditioning, but also the operational relevance of the learned mapping.

5.9 Stop Criteria and Simulation Termination

Each rollout terminates either when the expert guidance asserts the far-range stop flag or when a maximum simulation horizon is reached. The expert stop logic is persistence-based, as described in Chapter 3, so transient threshold crossings are not misclassified as successful completion.

For dataset quality and reproducibility, the stop condition is logged at each update. This enables several consistent policies during dataset assembly:

- samples collected after stop may be discarded, preventing near-zero terminal dynamics from dominating the dataset;
- failed rollouts, such as timeouts or simulator exceptions, may be retained and flagged, since they often contain informative difficult states;
- rollout-level success statistics can be computed consistently across expert and surrogate experiments.

This termination handling is particularly relevant for long-horizon guidance learning. If post-stop data were retained indiscriminately, the dataset would become increasingly dominated by trivial near-equilibrium samples, which would distort both training and evaluation.

5.10 Dataset Validation

Before training, the assembled dataset is subjected to deterministic sanity checks intended to detect silent pipeline failures. These checks include:

- **shape and finiteness checks**: verify that X_c and Y_c have the expected dimensions and remove non-finite rows;
- **feature-distribution checks**: inspect histograms and clipping statistics of $\mathbf{x}_{\text{clip}}$ to detect severe out-of-distribution behavior or preprocessing anomalies;
- **label-statistics checks**: inspect the distribution of $\|\mathbf{y}\|$ and the thrust-on fraction, and verify that target semantics match the intended training objective;

- **frame and time-consistency checks**: verify, through spot checks, that RTN conversion and time alignment reconstruct the expert command correctly from the stored logs.

These checks are critical in practice. In a pipeline with multiple scripts, generated artifacts, and evolving Simulink models, silent data corruption is more likely to arise from preprocessing mismatches or logging inconsistencies than from the learning algorithm itself. Deterministic validation therefore plays the role of a lightweight but essential integrity layer before training is launched.

5.11 Chapter Summary

This chapter presented the dataset-generation pipeline used to train the neural surrogate policy. The main elements of the pipeline are:

1. rollout generation in the Simulink closed-loop environment using the expert guidance law;
2. extraction of consistent state–action logs from compact simulator signals;
3. construction of history-stacked RTN features and normalized RTN command labels consistent with the deployed surrogate contract;
4. storage of the dataset in chunked artifacts together with frozen scaler statistics and deterministic split logic;
5. filtering, balancing, and validation procedures designed to improve robustness, prevent leakage, and preserve semantic consistency between training and deployment.

The key methodological point is that dataset generation is not treated as a passive by-product of simulation, but as a controlled and reproducible process that defines the learning problem seen by the surrogate. The next chapter uses these artifacts to describe the training procedure and the resulting comparison between expert and neural policy in open-loop and closed-loop operation.

Chapter 6

Expert Guidance vs Neural Surrogate: Results and Iterative Development

This chapter compares the model-based expert guidance law of Chapter 3 against the neural surrogate policy defined in Chapters 4 and 5. The comparison is carried out at two complementary levels. First, the surrogate is evaluated at the *action level*, by comparing its commanded acceleration against the expert command on the same state. Second, it is evaluated at the *trajectory level*, by assessing whether the closed-loop rollout remains stable and reaches the far-range handover condition under the state distribution induced by the surrogate itself.

The central claim of this chapter is that, for long-horizon low-thrust rendezvous, acceptable open-loop imitation is a necessary but not sufficient condition for reliable closed-loop guidance. The relevant failure mechanism is not only one-step approximation error, but the mismatch between the state distribution used during supervised training and the state distribution later induced by the deployed surrogate policy [15, 1]. For this reason, the chapter does not only report a final comparison: it also reconstructs the iterative development path that led from the first semantically consistent behavioral-cloning baseline to the strongest deployed surrogate configuration obtained in this thesis, and finally to the last semantic rerun performed to clarify the interpretation of the later post-aggregation degradation. Detailed numerical breakdowns, dataset audits, and iteration-by-iteration artifacts are not reported in full in the main text. The emphasis here is instead placed on the ranking of the main development stages, on the technical structure of the comparison, and on the distinction between the best operational configuration achieved in closed loop and the final semantic rerun, whose main value lies in clarifying and correcting the retraining problem.

6.1 Comparison Goals and Experimental Protocol

Let $\mathbf{s}_k$ denote the physical simulator state at guidance update k, and let

$$\mathbf{x}_k = \Phi_K(\mathbf{s}_{k-K+1}, \ldots, \mathbf{s}_k) \tag{6.1}$$

be the history-stacked surrogate feature vector introduced in Chapter 4, with $K = 5$. Let the expert policy be denoted by

$$\pi_E : \mathbf{s}_k \mapsto \mathbf{a}_E^R(k), \tag{6.2}$$

where $\mathbf{a}_E^R(k) \in \mathbb{R}^3$ is the commanded acceleration in the chaser RTN frame.

The learned surrogate is not used in deployment as a bare regressor. Rather, the deployed policy is the composition

$$\tilde{\pi}_\theta = \mathcal{W} \circ \pi_\theta \circ \mathcal{P}, \tag{6.3}$$

where:

- $\mathcal{P}$ denotes deterministic preprocessing (history stacking, normalization, clipping);
- π_θ denotes the trained neural network;
- $\mathcal{W}$ denotes the deterministic inference-time wrapper that maps the raw network output to the applied physical command.

This distinction is essential for interpretation. The object compared against the expert in closed loop is not only the neural network weights θ, but the full deployed module $\tilde{\pi}_\theta$.

6.1.1 Open-Loop Evaluation

In open-loop evaluation, the surrogate is queried on feature vectors logged from expert rollouts, and its predicted command is compared against the expert command computed on the same state sequence. Formally, the comparison is performed on samples drawn from the expert-induced state distribution

$$\mathbf{s}_k \sim d^{\pi_E}. \tag{6.4}$$

The open-loop surrogate output is therefore evaluated as

$$\mathbf{a}_{\mathrm{NN}}^R(k) = \tilde{\pi}_\theta(\mathbf{s}_k), \qquad \mathbf{s}_k \sim d^{\pi_E}. \tag{6.5}$$

This regime is fundamentally an interpolation test on the demonstrated manifold. It is appropriate for checking semantic consistency across:

- feature construction in the chaser RTN frame,
- history stacking and temporal ordering,
- frozen normalization statistics,
- output interpretation after export and deployment.

However, open-loop agreement does not directly establish whether the surrogate can preserve the trajectory near the expert manifold once it becomes the command generator itself.

6.1.2 Closed-Loop Evaluation

In closed-loop evaluation, the applied command is produced online by the deployed surrogate policy. The simulator evolves according to

$$\mathbf{s}_{k+1} = F(\mathbf{s}_k, \tilde{\pi}_\theta(\mathbf{s}_k)) \,, \tag{6.6}$$

so that the visited-state distribution becomes

$$\mathbf{s}_k \sim d^{\tilde{\pi}_\theta}. \tag{6.7}$$

This distinction is the technical core of the problem. Even if $\tilde{\pi}_\theta(\mathbf{s}) \approx \pi_E(\mathbf{s})$ on d^{π_E}, the rollout generated by $\tilde{\pi}_\theta$ may progressively move toward regions of the state space that were weakly represented during training. Once this occurs, the approximation error is no longer evaluated on the same distribution on which the supervised loss was minimized.

For fair action-level comparison under this setting, the expert command is still recomputed on the *surrogate-visited states*. That is, along a closed-loop surrogate rollout, both

$$\mathbf{a}^R_{\mathrm{NN}}(k) = \tilde{\pi}_\theta(\mathbf{s}_k), \qquad \mathbf{a}^R_E(k) = \pi_E(\mathbf{s}_k) \tag{6.8}$$

are evaluated on the same $\mathbf{s}_k \sim d^{\tilde{\pi}_\theta}$. This makes it possible to distinguish rollout failure from mere differences in the visited trajectory.

Because the surrogate input is history-stacked, a short deterministic warm-up phase is enforced before the neural command is allowed to drive the plant. During this phase, the plant is driven by the expert or by a safe fallback consistent with the deployment logic described in Chapter 4. After warm-up, the active command generator is switched to $\tilde{\pi}_\theta$.

6.1.3 Supervised Training Objective and Distribution Shift

The behavioral-cloning stage optimizes the surrogate on samples drawn from the expert distribution. Abstractly, the supervised objective can be written as

$$\theta^\star = \arg\min_\theta \mathbb{E}_{\mathbf{s}\sim d^{\pi_E}} \left[\ell(\tilde{\pi}_\theta(\mathbf{s}), \pi_E(\mathbf{s}))\right], \tag{6.9}$$

where $\ell(\cdot,\cdot)$ denotes a suitable action-level loss.

Equation (6.9) clarifies the limitation of pure behavioral cloning in sequential guidance. The objective is minimized on d^{π_E}, whereas deployment occurs under $d^{\tilde{\pi}_\theta}$. The relevant issue is therefore not only the magnitude of ℓ on the training distribution, but the discrepancy between the expert-induced and learner-induced state distributions, which in sequential decision problems can lead to compounding error over the rollout horizon [15, 1]:

$$d^{\tilde{\pi}_\theta} \neq d^{\pi_E}. \tag{6.10}$$

6.1.4 Comparison Metrics

At action level, the main quantity of interest is directional agreement between expert and surrogate. Let $\mathbf{a}_E^R(k)$ and $\mathbf{a}_{\mathrm{NN}}^R(k)$ denote the expert and deployed-surrogate commands in the chaser RTN frame. Directional agreement is quantified through cosine similarity:

$$\cos(k) \triangleq \frac{\mathbf{a}_E^R(k)^\top \mathbf{a}_{\mathrm{NN}}^R(k)}{\|\mathbf{a}_E^R(k)\|\,\|\mathbf{a}_{\mathrm{NN}}^R(k)\| + \varepsilon_c}. \tag{6.11}$$

Magnitude behavior is characterized through the norms

$$m_E(k) \triangleq \|\mathbf{a}_E^R(k)\|, \qquad m_{\mathrm{NN}}(k) \triangleq \|\mathbf{a}_{\mathrm{NN}}^R(k)\|, \tag{6.12}$$

and, when useful for interpretation, through the ratio

$$r_m(k) \triangleq \frac{m_{\mathrm{NN}}(k)}{m_E(k) + \varepsilon_m}. \tag{6.13}$$

At trajectory level, the relevant object is the rollout outcome. Let $\mathcal{H}$ denote the far-range handover set, and let

$$S_i = \begin{cases} 1, & \exists\, k \le K_{\max} \text{ such that } \mathbf{s}_k^{(i)} \in \mathcal{H}, \\ 0, & \text{otherwise}, \end{cases} \tag{6.14}$$

be the success indicator of rollout i. The empirical success rate over a campaign of N rollouts is

$$\hat{p}_{\mathrm{succ}} = \frac{1}{N}\sum_{i=1}^{N} S_i. \tag{6.15}$$

In addition to success, the analysis tracks time-to-handover for successful rollouts and the qualitative class of failure for unsuccessful ones. This is necessary because two surrogates with similar one-step action metrics may still induce very different closed-loop regimes.

6.2 Iterative Development of the Surrogate

The final picture reported in this thesis did not emerge from a single training cycle. Rather, it was obtained through a sequence of iterations whose purpose was to identify which component of the pipeline was responsible for the observed gap between open-loop plausibility and closed-loop reliability.

This iterative development is technically relevant because it shows that the difficulty of the problem is not reducible to offline regression accuracy. What had to be aligned was the full chain

$$\begin{array}{c} \text{dataset semantics} \\ \Downarrow \\ \text{training target} \\ \Downarrow \\ \text{exported artifact} \\ \Downarrow \\ \text{deployed command interpretation} \\ \Downarrow \\ \text{closed-loop behavior} \end{array} \tag{6.16}$$

A mismatch at any point of the chain can make the learned policy appear worse than it actually is, or apparently acceptable in open loop while still failing in closed loop.

Table 6.1 summarizes the development path at the level of technical intent and observed regime. The following subsections formalize the role of each stage more precisely.

6.2.1 Frame-Consistent Behavioral-Cloning Baseline

The first meaningful baseline was obtained only after enforcing semantic consistency between offline dataset generation and deployed inference. This step required alignment of:

- feature construction in the chaser RTN frame,
- history stacking and temporal ordering,

Table 6.1: Traceability summary of the main surrogate-development iterations discussed in this chapter. The table highlights the distinction between the strongest deployed configuration obtained in the thesis and the final semantic rerun, whose main contribution was to clarify the retraining problem rather than to establish a superior closed-loop rendezvous policy.

Iter.	Stage	Technical intervention	Targeted issue	Observed outcome / interpretation
0	Frame-consistent BC baseline	Consolidation of feature-frame, label-frame, normalization, and deployed output semantics.	Remove trivial semantic inconsistency and obtain a meaningful supervised baseline.	Open-loop interpolation became meaningful, but closed-loop reliability remained insufficient due to policy-induced distribution shift.
1	First aggregation stage	Initial DAgger-style relabeling on surrogate-visited states and retraining on aggregated chunks.	Reduce mismatch between d^{π_E} and d^{π_θ}.	Meaningful improvement over the pure frame-consistent behavioral-cloning baseline. Aggregation was therefore beneficial at this stage.
2	Deployment regularization of the aggregated policy	Deterministic input clamp, constrained output handling, deadband, and confidence-based airbag attenuation in the inference wrapper.	Bound extrapolation and regularize the improved policy in closed loop.	Best operational closed-loop result of the thesis. The surrogate remained only moderately aligned with the expert and conservative in magnitude, but reached the strongest bounded and interpretable operating regime of the study.
3	Later post-constraint aggregation attempts	Additional aggregation and retraining on rollouts generated by the already constrained policy.	Test whether further aggregation could improve the stabilized aggregated surrogate.	Did not robustly supersede the previous best configuration. Some runs degraded, but interpretation became sensitive to target semantics, artifact integrity, and retraining consistency.
4	Final corrected semantic rerun	Rebuilt full-contract base dataset, regenerated corrected aggregated chunks, enforced explicit direction-plus-magnitude target semantics, and retrained on a semantically uniform merged corpus.	Remove residual ambiguity due to inconsistent target contract in later post-constraint retraining.	Learning-side metrics improved markedly, confirming that semantic contamination had been a major confounder. However, no conclusive closed-loop gain beyond the earlier best deployed configuration was established.

- frozen input normalization,
- output interpretation as normalized RTN command.

Only after this alignment did the behavioral-cloning baseline become interpretable as a valid approximation of the expert policy. In open loop, the surrogate then behaved as a meaningful interpolant on the expert-generated manifold. This established that the supervised mapping itself was not degenerate.

However, once the surrogate was inserted in closed loop, the rollout still degraded. This observation is technically important because it separates two different explanations:

1. a *semantic inconsistency explanation*, in which the network fails because the deployed policy is not the policy that was actually trained;
2. a *distribution-shift explanation*, in which the trained mapping is locally meaningful on d^{π_E} but cannot preserve the trajectory near that distribution once deployed.

The observed behavior after semantic consolidation supported the second explanation as the dominant one.

6.2.2 First Aggregation Stage

Once distribution shift had been identified as the dominant mechanism, the natural next step was dataset aggregation. At an abstract level, the corresponding update can be described as

$$\mathcal{D}_{j+1} = \mathcal{D}_j \cup \left\{ (\mathbf{s}, \pi_E(\mathbf{s})) \ : \ \mathbf{s} \sim d^{\tilde{\pi}_{\theta_j}} \right\}, \tag{6.17}$$

followed by retraining on the aggregated dataset.

The logic of (6.17) is straightforward: if the surrogate is later evaluated on states drawn from $d^{\tilde{\pi}_\theta}$, then the training set should be expanded to include expert labels on that same distribution. Conceptually, this was the correct corrective direction to test.

In practice, the first aggregation stage produced a meaningful improvement over the pure frame-consistent behavioral-cloning baseline. This point is important for the interpretation of the whole thesis. The development history does not support a blanket conclusion that aggregation was ineffective from the outset; on the contrary, the first aggregated retraining stage yielded a surrogate that behaved better than the original baseline and therefore confirmed that bringing expert labels closer to the learner-induced distribution could be beneficial.

At the same time, this early improvement did not by itself solve the closed-loop reliability problem. The aggregated surrogate remained sensitive to out-of-distribution excursions and still required explicit deployment-side regularization before it could become a robustly interpretable controller.

6.2.3 Deployment Regularization of the Aggregated Policy

After the first aggregation stage had improved the baseline, the development priority shifted to making that improved policy safer and more stable in deployment. The objective was not to change the training target further, but to regularize the way in which the learned command was applied online.

The first measure was deterministic input clamping. Let

$$\mathbf{x}_{\text{norm}} = (\mathbf{x}_k - \boldsymbol{\mu}_X) \oslash \boldsymbol{\sigma}_X \tag{6.18}$$

denote the feature vector standardized with frozen training-set statistics, where $\oslash$ denotes component-wise division. The deployed network input is then

$$\mathbf{x}_{\text{clip}} = \text{clip}(\mathbf{x}_{\text{norm}}, -8, +8)\,. \tag{6.19}$$

This limits the effect of rare outliers and bounds the effective operating range of the regressor when queried on poorly represented states.

Let $\mathbf{y}_k = \pi_\theta(\mathbf{x}_{\text{clip}}) \in [-1,1]^3$ denote the raw network output. Define

$$u_k = \|\mathbf{y}_k\|, \qquad \hat{\mathbf{u}}_k = \frac{\mathbf{y}_k}{u_k + \varepsilon_u}. \tag{6.20}$$

The deployed command is not applied directly from $\mathbf{y}_k$. Instead, the wrapper constructs

$$\mathbf{a}^R_{\text{NN}}(k) = a_{\max}(k)\, \alpha_k\, \hat{\mathbf{u}}_k, \tag{6.21}$$

where $\alpha_k \in [0,1]$ is a deterministic thrust fraction produced after constrained output handling, deadbanding, magnitude mapping, and confidence-based attenuation.

In compact operator form, the deployed command law is

$$\mathbf{a}^R_{\text{NN}}(k) = \mathcal{W}(\mathbf{x}_{\text{clip}}, \mathbf{y}_k, a_{\max}(k))\,. \tag{6.22}$$

The purpose of (6.19)–(6.22) is not to improve training loss. Their role is to regularize deployment. In practice, these measures changed the dominant regime from fragile closed-loop behavior to a much more stable and interpretable rollout class. Within the development sequence reconstructed here, this stage produced the best operational result of the thesis: the surrogate was no longer merely a promising learned regressor, but a bounded deployed controller that could be compared meaningfully against the expert in closed loop.

6.2.4 Post-Constraint Aggregation Attempts

Once the constrained aggregated policy had become the best available deployed configuration, aggregation was revisited. This chronology is important. The later retraining attempts were not applied to the original unstable behavioral-cloning baseline, but to a policy that had already been improved by the first aggregation stage and then regularized through deployment-side safety measures.

These post-constraint aggregation attempts were conceptually well motivated: if the constrained policy now generated cleaner rollouts, then the corresponding visited-state distribution might yield more meaningful relabeling and better aggregated retraining. In other words, the goal was to test whether a more stable deployed policy could bootstrap a better second generation of aggregation data.

In practice, these later aggregation attempts did not robustly supersede the previous best configuration. Some runs degraded, others failed to produce a clear gain, and the interpretation of the results became increasingly sensitive to target semantics, artifact integrity, and the exact contract used by the aggregated labels. This is the stage at which the main ambiguity of the final part of the campaign emerged: the observed degradation could not be attributed uniquely to the intrinsic limitation of further aggregation, because the semantic integrity of the post-constraint aggregated targets was not yet guaranteed.

6.2.5 Final Corrected Semantic Rerun

The final experimental campaign was designed to answer a precise question: whether the degradation observed in the later post-constraint aggregation attempts should be interpreted as a genuine limitation of those further retraining steps, or whether it had been confounded by a semantic inconsistency in the target contract of the aggregated chunks and in the merged retraining corpus.

To remove this ambiguity, the final rerun imposed an explicit full-contract target semantics of the form

$$\mathbf{Y} = \hat{\mathbf{u}}_E^R \, \text{mag\_frac}, \tag{6.23}$$

where $\hat{\mathbf{u}}_E^R$ is the unit expert command direction in the chaser RTN frame and mag_frac $\in [0,1]$ is the expert command magnitude fraction relative to the available acceleration authority. Under this contract,

$$\|\mathbf{Y}\|_2 = \text{mag\_frac}. \tag{6.24}$$

The final rerun therefore involved three coupled steps:

1. correction of the aggregation-chunk builder so that the new DAgger chunks explicitly respected the full-contract target definition;

2. rebuilding of the base dataset under the same full-contract semantics, rather than merging corrected aggregated chunks with a semantically heterogeneous legacy base corpus;

3. retraining on the resulting semantically uniform merged dataset.

This step is central to the interpretation of the final results. The later post-constraint degradation could not be treated as a definitive verdict against further aggregation until the target contract itself had been made explicit, uniform, and auditable. The final rerun closed that ambiguity. Its importance, however, lies primarily in clarifying the semantic correctness of the retraining problem. It does not replace the earlier best operational configuration as the main deployed result of the thesis.

6.3 Open-Loop Results

The open-loop experiments confirm that, once semantic consistency is enforced, the surrogate behaves as a meaningful approximation of the expert command on expert-generated trajectories. In formal terms, the deployed module $\tilde{\pi}_\theta$ acts as a nontrivial interpolant on samples drawn from d^{π_E}.

This conclusion is important, but it must be interpreted correctly. Open-loop agreement is not the final objective of the chapter; rather, it is a validity check showing that the learned mapping is technically meaningful once feature construction, temporal ordering, normalization, and output interpretation are aligned. In this sense, the open-loop campaign establishes that the surrogate is not failing for trivial semantic reasons.

Representative diagnostics retained for the strongest deployed configuration show moderate but clearly non-degenerate action agreement on expert-generated trajectories. For thrust-active samples, the open-loop cosine similarity is ≈ 0.331 in mean and ≈ 0.510 in median, while the magnitude ratio $\|\mathbf{a}^R_{\mathrm{NN}}\|/(\|\mathbf{a}^R_E\| + \varepsilon_m)$ is ≈ 0.67 in mean and ≈ 0.43 in median. These values do not indicate expert-equivalent action matching, but they do confirm that the surrogate learned a meaningful command map on the expert manifold. In the same retained diagnostic run, the clipped normalized inputs remain well inside the clamp boundaries, with

$$x_{\mathrm{norm,clip}} \in [-1.861,\ 1.681],$$

which is consistent with interpolation rather than strong extrapolation.

Directional agreement remains the most informative action-level indicator. In long-horizon low-thrust guidance, moderate magnitude mismatch can often be interpreted as a difference in aggressiveness, whereas persistent directional bias alters the secular evolution of the trajectory and therefore has a much stronger

effect on the subsequent state distribution. For this reason, the main open-loop role of the surrogate is not to reproduce every command exactly, but to remain directionally coherent and physically bounded on the expert manifold.

Across the development path reconstructed here, the open-loop interpretation remains consistent: semantic consolidation is necessary to obtain a meaningful policy approximation, aggregation can improve that approximation, and the final semantic rerun confirms that the full-contract target can be learned much more coherently once the dataset is made semantically uniform. None of these observations, however, is sufficient on its own to establish reliable closed-loop guidance.

6.4 Closed-Loop Results

The closed-loop analysis remains the decisive part of the comparison, because the central limitation of the surrogate emerges only when the visited-state distribution becomes policy-induced. Even a much better supervised fit can remain insufficient if the surrogate still accumulates directional error under propagation.

The development sequence reported in this thesis supports a more nuanced conclusion than a simple "aggregation works" or "aggregation fails" verdict. The first aggregation stage did improve the original frame-consistent behavioral-cloning baseline. Moreover, the best operational result of the entire study was obtained when that improved policy was combined with deterministic deployment regularization, including clamp, constrained output handling, deadband, and airbag attenuation. This combined configuration yielded the most stable and interpretable closed-loop regime reached in the project.

Representative diagnostics for that best deployed configuration are reported in Table 6.2. The table does not claim to summarize a fully homogeneous Monte Carlo campaign; rather, it reports two retained closed-loop rollout diagnostics that were considered representative of the strongest operational regime achieved in the thesis.

These diagnostics show a consistent pattern. First, the surrogate remains directionally aligned with the expert in closed loop, but only at a moderate level: the mean thrust-direction cosine is 0.329 and 0.340 in the two retained runs, while the median is 0.397 and 0.475. Second, the applied command magnitude is systematically conservative with respect to the expert, with mean magnitude ratios of 0.41 and 0.31, and median ratios of 0.45 and 0.30. This is fully consistent with the interpretation of the best deployed configuration as a bounded and operationally usable controller, but not an expert-equivalent one.

The additional diagnostics are also informative. In both retained closed-loop runs, the applied command remains almost perfectly aligned with the raw neural

Table 6.2: Representative diagnostics for the best deployed surrogate configuration obtained in this work, corresponding to the first aggregation stage combined with deterministic deployment regularization. Closed-loop values are computed on surrogate-visited states and thrust-active samples.

Metric	**Case 1**	**Case 2**
Closed-loop mean cosine (thrust)	0.329	0.340
Closed-loop median cosine (thrust)	0.397	0.475
Closed-loop mean magnitude ratio	0.41	0.31
Closed-loop median magnitude ratio	0.45	0.30

command, with

$$\cos(\mathbf{a}_{\text{applied}}, \mathbf{a}_{\text{NN}}) \approx 1,$$

which indicates that the wrapper is not materially rotating the neural command direction. Its role is instead to regularize the command magnitude and to bound the controller near extrapolative regions. At the same time, the normalized inputs do reach the clamp boundaries in closed loop,

$$x_{\text{norm,clip}} \in [-8,\ 8],$$

whereas the unclipped normalized state reaches much larger excursions. This confirms that the best deployed regime is already operating in a distribution-shifted setting and that deterministic regularization is an integral part of the result.

The role of this result is important. It shows that the most effective progress of the study did not come from supervised training alone, nor from deployment constraints alone, but from their combination: a first aggregation stage that improved the original baseline, followed by deterministic deployment regularization that made the improved policy usable in closed loop.

Later post-constraint aggregation attempts did not robustly improve upon that best configuration. Some of those runs degraded, but their interpretation was initially ambiguous because the aggregated target contract was not yet fully controlled. The final semantic rerun was introduced precisely to resolve this ambiguity.

A benchmark campaign was then attempted to compare the best deployed configuration and the final retrained full-contract model automatically over multiple closed-loop cases. However, that evaluation harness produced suspicious outputs, including incorrect case identifiers and terminal quantities that were too uniform across cases to be considered fully trustworthy. For this reason, the automatic A/B benchmark was not considered reliable enough to support a final comparative claim and is not used here as conclusive evidence.

Table 6.3: Training-side metrics for post-constraint retraining before and after enforcing a semantically uniform full-contract dataset. The mixed merged corpus combines corrected aggregated chunks with the historical base dataset; the uniform merged corpus combines corrected aggregated chunks with the rebuilt full-contract base dataset.

Retraining corpus	Best val. loss	Approx. test loss	Approx. test cosine	Approx. test magMAE	Approx. test normCorr
Semantically mixed merged corpus	6.64×10^{-1}	9.28×10^{-1}	0.436	0.235	0.548
Semantically uniform full-contract merged corpus	1.4937×10^{-1}	1.6337×10^{-1}	0.8937	0.15848	0.5998

The correct conclusion is therefore asymmetric. The best deployed surrogate configuration can be promoted as the strongest operational result obtained in the thesis. By contrast, the final semantic rerun can be promoted confidently on learning-side grounds, but not as a superior closed-loop result beyond that earlier best configuration.

6.5 Training-Side Effect of the Final Semantic Rerun

The clearest positive result of the final campaign appears on the learning side. Before rebuilding the base corpus, a diagnostic retraining had been performed on a semantically mixed merged dataset obtained by combining corrected aggregated chunks with the historical base dataset. That experiment was informative but not definitive, because the merged corpus combined two distinct target semantics and therefore could not serve as a clean final test of the later post-constraint aggregation stage.

Once the base corpus was rebuilt under the same full-contract target contract and merged with the corrected aggregated chunks, retraining quality improved substantially. Table 6.3 summarizes this contrast.

The improvement is not marginal. The loss decreases drastically, directional agreement on the held-out test subset increases sharply, and magnitude reconstruction also improves. This confirms that the degradation seen in the semantically mixed post-constraint retraining could not be interpreted as a final verdict against the later aggregation stage itself, because the retraining corpus had been semantically contaminated.

At the same time, this result must be interpreted at the correct level. Table 6.3 shows that semantic target consistency matters decisively for learning a meaningful full-contract mapping. It does not show that the resulting model surpasses the earlier best deployed configuration in closed loop. That stronger claim would

require a reliable trajectory-level comparative campaign, which is not available here.

6.6 Discussion

The comparison between expert and surrogate supports five main conclusions.

First, **open-loop imitation is necessary but not sufficient**. A surrogate can reproduce the expert with meaningful fidelity on d^{π_E} and still fail to preserve acceptable rollout behavior on $d^{\tilde{\pi}_\theta}$. This is the classical limitation of behavioral cloning in sequential decision systems and is fully consistent with (6.10) [1].

Second, **a first aggregation stage did help**. The development record does not support the claim that aggregation was ineffective from the start. On the contrary, the first DAgger-style retraining stage improved the frame-consistent behavioral-cloning baseline and therefore confirmed that moving supervision closer to the learner-induced distribution can yield a meaningful benefit.

Third, **the best operational result was obtained when that improved policy was regularized at deployment**. Clamp, constrained output logic, deadband, and airbag do not solve the learning problem in isolation, but they define the deployed controller that is actually evaluated:

$$\tilde{\pi}_\theta = \mathcal{W} \circ \pi_\theta \circ \mathcal{P}. \tag{6.25}$$

In the development sequence reconstructed here, this combination of early aggregation benefit and deterministic deployment regularization produced the strongest closed-loop regime achieved in the thesis. The retained diagnostics show that this regime remains only moderately aligned with the expert in direction and systematically conservative in command magnitude, but still clearly represents the best bounded operating point reached in the study.

Fourth, **later post-constraint aggregation attempts did not robustly supersede that best configuration**. Some later runs degraded, but their interpretation was not initially clean because target semantics and artifact integrity had become part of the problem. The final semantic rerun clarified this point by showing that once the target contract is made fully uniform, learning-side metrics improve sharply.

Fifth, **improved full-contract learning does not yet imply a superior rendezvous-grade closed-loop policy**. The final semantic rerun demonstrates that the retraining problem can be made much more coherent and that the network can learn the intended direction-plus-magnitude contract far better than in the semantically mixed case. Yet the currently available closed-loop evidence is still insufficient to claim that this final retrained model surpasses the earlier best deployed configuration. In other words, semantic repair of the target contract is

necessary for a clean interpretation of later aggregation, but it is not by itself sufficient to establish a stronger operational result.

Taken together, these observations support the main interpretation of the thesis. The relevant gap is not simply between expert and neural network. It is between a supervised approximation that can be improved and semantically repaired on the learning side, and a deployed surrogate that must still remain reliable under the state distribution it induces itself.

6.7 Chapter Summary

This chapter compared the model-based expert guidance law and the neural surrogate at both action level and trajectory level, while reconstructing the iterative development path that led from the first frame-consistent behavioral-cloning baseline to the strongest deployed surrogate configuration and finally to the last semantic rerun.

The results show that semantic consolidation is sufficient to obtain a meaningful open-loop surrogate, but not sufficient to guarantee reliable closed-loop guidance. A first aggregation stage then produced a real improvement over the pure behavioral-cloning baseline, confirming that distribution-aware relabeling was directionally useful. The strongest operational result of the thesis was obtained when that improved policy was combined with deterministic deployment regularization—input clamp, constrained output handling, deadband, and confidence-based airbag attenuation—yielding the most stable and interpretable closed-loop regime reached in the project.

Representative diagnostics of that best configuration show moderate but consistently positive closed-loop directional agreement with the expert, with thrust-direction cosine similarity around 0.33 in mean and roughly 0.39–0.47 in median across the retained rollout diagnostics, together with systematically conservative magnitude ratios well below unity. This confirms that the best deployed surrogate is not expert-equivalent, but rather the strongest bounded operational compromise achieved in the study.

Subsequent post-constraint aggregation attempts did not robustly improve upon that best configuration. The final semantic rerun then removed a major ambiguity affecting the interpretation of those later experiments. By rebuilding a semantically uniform full-contract base dataset, regenerating corrected aggregated chunks, and retraining on a consistent merged corpus, it showed that semantic target consistency materially improves learning-side performance. However, this does not yet provide conclusive evidence of a superior rendezvous-grade closed-loop policy beyond the earlier best deployed configuration. The final conclusion is therefore twofold.

Scientifically, the chapter confirms that the main difficulty of imitation-based low-thrust guidance lies in closed-loop robustness under policy-induced distribution shift. Methodologically, it shows that semantic target consistency is necessary for interpreting later aggregation correctly, yet is not by itself sufficient to establish a stronger deployed result than the best configuration reached earlier in the study.

Chapter 7

Conclusions and Future Work

This thesis addressed the problem of autonomous far-range rendezvous under low-thrust actuation, with particular emphasis on the tension between guidance quality, long-horizon closed-loop robustness, and deployability in an onboard-oriented architecture. The work was structured around a deliberately layered methodology: first, a model-based expert guidance law was defined and validated as a reference policy; second, a data-generation and preprocessing pipeline was constructed to produce semantically consistent supervision; third, a compact neural surrogate was trained and integrated as a drop-in command generator within the same simulation environment used by the expert.

A first contribution of the thesis is therefore methodological. Rather than treating learning as an isolated regression problem, the surrogate was developed within a traceable guidance workflow in which feature construction, normalization, deployment semantics, warm-up logic, and evaluation protocol were all made consistent with the final closed-loop use case. This aspect proved essential. In long-horizon low-thrust rendezvous, small inconsistencies between offline training conventions and online inference semantics can invalidate the practical meaning of an otherwise well-performing supervised model.

A second contribution is the construction of a model-based expert baseline suitable for large-scale data generation and comparative assessment. The expert policy provided a physically interpretable reference mapping from relative orbital state information to commanded acceleration, and enabled the generation of rollout-level evidence across diverse initial conditions. This made it possible to evaluate the learned policy not only in terms of one-step agreement, but in terms of actual guidance behavior and mission-oriented convergence properties.

The central result of the thesis is that acceptable open-loop imitation does not

automatically translate into reliable closed-loop rendezvous guidance. When evaluated on expert-generated trajectories, the surrogate was able to reproduce relevant aspects of the expert command with meaningful action-level agreement. However, once inserted online in the loop, approximation error interacted with distribution shift and long-horizon propagation, producing a much more demanding operating regime. This confirms a key point for learning-based guidance: in sequential decision problems, the relevant failure mechanism is often not the instantaneous regression error itself, but the progressive accumulation of state-distribution mismatch induced by the learned controller.

For this reason, the thesis did not stop at a single behavioral-cloning benchmark. The surrogate development was iterated through multiple stages including semantic consistency checks, deployment validation, stabilization measures, and renewed aggregation attempts. This process showed that part of the performance gap could be mitigated through engineering decisions such as robust preprocessing, bounded inference behavior, and safer command handling. These measures did not fully eliminate the gap with respect to the expert, but they were sufficient to obtain a more stable and interpretable learned baseline. In this sense, the thesis contributes not only a final model, but also a documented development path explaining which interventions were necessary to move from nominal imitation to a defensible integrated surrogate.

From a broader perspective, the results support a cautious but constructive view of supervised learning for autonomous rendezvous guidance. A compact neural policy can be trained to approximate a model-based expert and can be embedded cleanly into a deterministic simulation architecture. This is promising from the standpoint of computational efficiency and software integration. At the same time, the present results indicate that for long-horizon low-thrust guidance, pure supervised imitation alone is generally insufficient to guarantee the same level of convergence reliability as the expert policy. The main challenge is not only function approximation, but robustness under the policy-induced state distribution.

The main limitations of the present work follow directly from this conclusion. First, the surrogate architecture remains memory-limited, since temporal context is provided through a short fixed history window rather than through an explicitly recurrent representation of long-horizon guidance evolution. Second, the training distribution, although diversified and iteratively enriched, still only approximates the set of states visited in closed loop by the learned controller. Third, the evaluation has focused primarily on nominal and controlled comparative conditions, complemented by perturbation models representative of realistic LEO operations. This is appropriate for establishing traceable baseline behavior, although it does not yet exhaust the full range of operational uncertainties relevant to flight applications. Finally, the learned policy has been treated as a guidance surrogate within a simulation-centered validation workflow, rather than as a component already

qualified for embedded autonomy under strict real-time and verification constraints. These limitations naturally define the most relevant directions for future work. A first direction is to improve the handling of distribution shift through more systematic dataset aggregation, including more selective relabeling strategies, curriculum-based rollout enrichment, and explicit prioritization of failure states. A second direction is architectural: recurrent or sequence-aware models could provide a richer representation of guidance context than fixed-window stacking, potentially improving consistency over long horizons. A third direction is the inclusion of confidence-aware or safety-filtered inference logic, so that the learned policy can be bounded or overridden when it operates outside the distribution in which it was trained. This would be especially relevant in mixed autonomy schemes where a learned policy accelerates nominal guidance while a model-based supervisor preserves operational safety.

A further line of development concerns validation depth. The surrogate should be assessed under broader perturbation campaigns, including model mismatch, navigation uncertainty, thrust dispersion, and additional operational constraints. In parallel, the current workflow could be extended toward embedded implementation by measuring inference latency, memory footprint, determinism, and integration compatibility with representative onboard software environments. In this form, the present thesis can be seen as a necessary intermediate step between proof-of-concept learning and guidance software that is credible for mission use.

In conclusion, this thesis has shown that supervised learning can provide a meaningful surrogate approximation of a model-based far-range rendezvous guidance law, but also that the true difficulty emerges only when the learned policy is evaluated in closed loop over long low-thrust horizons. The work therefore highlights both the promise and the current limits of imitation-based guidance for autonomous rendezvous. Its main outcome is not the claim that a neural policy already replaces the expert, but the establishment of a rigorous workflow to design, integrate, diagnose, and critically assess such surrogates in a way that is technically transparent and extensible toward more robust future solutions.

Bibliography

[1] Stephane Ross, Geoffrey Gordon, and Drew Bagnell. «A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning». In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Vol. 15. Proceedings of Machine Learning Research. 2011, pp. 627–635 (cit. on pp. 2, 4, 50, 59, 62, 68, 72, 75, 85).

[2] Anastassios E. Petropoulos. «Low-Thrust Orbit Transfers Using Candidate Lyapunov Functions with a Mechanism for Coasting». In: *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*. 2004, pp. 748–762. DOI: 10.2514/6.2004-5089 (cit. on pp. 3, 31, 32).

[3] Anastassios E. Petropoulos. «Refinements to the Q-law for the Low-Thrust Orbit Transfers». In: *AAS/AIAA Space Flight Mechanics Conference*. Vol. 120. Advances in the Astronautical Sciences. AAS 05-162. 2005, pp. 963–982 (cit. on pp. 3, 31, 32).

[4] Jackson L. Shannon, Martin T. Ozimek, Justin A. Atchison, and Christine M. Hartzell. «Q-Law Aided Direct Trajectory Optimization of Many-Revolution Low-Thrust Transfers». In: *Journal of Spacecraft and Rockets* 57.4 (2020), pp. 672–682. DOI: 10.2514/1.A34586 (cit. on p. 3).

[5] Jackson L. Shannon, Martin T. Ozimek, Justin A. Atchison, and Christine M. Hartzell. «Rapid Design of High-Fidelity Low-Thrust Transfers to the Moon». In: *Journal of Spacecraft and Rockets* 59.5 (2022), pp. 1522–1535. DOI: 10.2514/1.A35177 (cit. on p. 3).

[6] M. J. H. Walker, B. Ireland, and J. Owens. «A Set of Modified Equinoctial Orbit Elements». In: *Celestial Mechanics* 36.4 (1985), pp. 409–419. DOI: 10.1007/BF01227493 (cit. on pp. 3, 9–11, 13, 34, 37).

[7] David A. Vallado. *Fundamentals of Astrodynamics and Applications*. 4th ed. Microcosm Press, 2013 (cit. on pp. 3, 34, 40).

[8] Howard D. Curtis. *Orbital Mechanics for Engineering Students*. 4th ed. Elsevier, 2020. ISBN: 9780128240250 (cit. on pp. 9, 10).

[9] Roger A. Broucke and Paul J. Cefola. «On the Equinoctial Orbit Elements». In: *Celestial Mechanics* 5 (1972), pp. 303–310. DOI: 10.1007/BF01228432 (cit. on pp. 9, 11, 13).

[10] *IERS Conventions (2010)*. Tech. rep. IERS Technical Note No. 36. International Earth Rotation and Reference Systems Service (IERS), 2010 (cit. on p. 18).

[11] Sanjeev Narayanaswamy and Christopher J. Damaren. «Equinoctial Lyapunov Control Law for Low-Thrust Rendezvous». In: *Journal of Guidance, Control, and Dynamics* 46.4 (2023), pp. 781–795. DOI: 10.2514/1.G006662 (cit. on pp. 31, 32).

[12] Yang Gao. «Near-Optimal Very Low-Thrust Earth-Orbit Transfers and Guidance Schemes». In: *Journal of Guidance, Control, and Dynamics* 30.2 (2007), pp. 529–539. DOI: 10.2514/1.24836 (cit. on p. 40).

[13] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. *Layer Normalization*. 2016. arXiv: 1607.06450 [stat.ML] (cit. on p. 55).

[14] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. «Deep Sparse Rectifier Neural Networks». In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Vol. 15. Proceedings of Machine Learning Research. 2011, pp. 315–323 (cit. on p. 55).

[15] Stephane Ross and Drew Bagnell. «Efficient Reductions for Imitation Learning». In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Vol. 9. Proceedings of Machine Learning Research. PMLR, 2010, pp. 661–668 (cit. on pp. 62, 72, 75).

[16] Wassily Hoeffding. «Probability Inequalities for Sums of Bounded Random Variables». In: *Journal of the American Statistical Association* 58.301 (1963), pp. 13–30. DOI: 10.1080/01621459.1963.10500830 (cit. on p. 63).