POLITECNICO DI TORINO

MASTER of MECHANICAL ENGINEERING

MASTER's Degree Thesis

Collaborative Robotics and Collision Avoidance in Human-Robot Shared Workspaces

Candidate:

Kosar Akbari

Supervisor:

Prof. Stefano Mauro
(DIMEAS)

Co-Supervisors:

Matteo Melchiorre
Laura Salamina

October 2025

To my beloved family;
whose unwavering love, support, and encouragement have been my foundation throughout this journey.
Their confidence in me has been the driving force behind every step of this work.

Abstract

Collaborative robotics blends the precision of industrial manipulators with human dexterity in shared workspaces. In practice, safety dictates motion: power-and-force limiting and, especially, speed-and-separation monitoring (SSM) shape behavior to maintain provable human–robot clearances. Turning policy into control requires phrasing separation and interaction limits as bounds on position, velocity, and effort, then generating commands that remain feasible under sensing noise and kinematic constraints. For redundant manipulators, task-priority control with null-space projection achieves the primary end-effector objective while shaping posture, respecting joint limits, and accommodating collision-avoidance biases derived from perception. Robustness near kinematic singularities is provided by damped least squares (DLS) solved via singular-value decomposition (SVD), which attenuates ill-conditioned directions in real time. References are either a bounded Cartesian attractive velocity (interactive runs) or linear-segment-with-parabolic-blend (LSPB) trajectories (time-parameterized runs); both are executed under the same SVD-regularized DLS inverse kinematics (IK) and supervisory gating, while light Cartesian damping closes residual errors. Coupled with an explicit state-machine supervisor that gates approach, stop, hold, repel, and resume, these elements provide a principled path from safety policy to executable motion.

The thesis develops the modeling, algorithms, and implementation to realize that framework end-to-end: from velocity-field target acquisition (interactive) and LSPB tracking (time-parameterized), to a separate fixed tool center point (TCP) regime where redundancy alone is used to reshape posture, through collaborative operation that pauses motion inside a risk envelope and resumes only after persistent clearance, to a fixed-TCP regime where the arm reconfigures purely in the Jacobian null space to increase separation. Observability and transparency are emphasized: task-space singular values and condition numbers quantify nearness to singularity; linear manipulability and joint-saturation flags expose control effort; and a conflict metric reports alignment between tracking and avoidance. An acceptance radius and a time-based deadband for decisive stops yield reproducible gate behavior; a gentle orientation lock avoids wrist flips; and discrete-time

consistency is enforced by tying LSPB sampling to the physics step and performing projection/smoothing before integration (notably in the fixed-TCP case).

Empirically, the unified LSPB–DLS–SVD framework acquires targets without overshoot, halts and resumes predictably under SSM-like proximity events, and when the TCP is fixed, maintains negligible drift while redistributing motion across the redundant chain to maximize clearance. The result is an implementation-level account of how trajectory time-parameterization, SVD-regularized DLS IK, and null-space safety fields can be composed under explicit state-machine supervision to deliver interpretable, robust collision avoidance for collaborative manipulation in shared workspaces.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter establishes the conceptual foundations for collaborative robotics and collision avoidance in shared workspaces. The objective is to situate the problem within industrial practice, articulate the safety and control principles that govern motion in the presence of humans, and frame the sensing and modelling choices that make separation monitoring implementable at the control rates used in modern manipulators.

Collaborative robotics replaces rigid spatial segregation with coordinated human–robot activity within a common workspace. For practical deployment in human-occupied environments, robot motion must be readily interpretable by nearby workers, degrade gracefully under perception uncertainty, and remain within well-defined safety envelopes. Industrial practice distinguishes interaction regimes by how space and time are shared, ranging from fenced isolation, through coexistence and sequential collaboration, to cooperation and fully responsive collaboration in which both agents move concurrently and adapt in real time. As responsiveness increases, the demands on perception latency and controller update rate tighten, and the control system must revise motion online without sacrificing task performance or eroding safety margins.

Safety policy in shared workspaces is commonly structured along two complementary lines. Power-and-force limiting (PFL) constrains the consequences of contact by bounding forces, torques, velocities, or momentum. Speed-and-separation monitoring aims to prevent contact by regulating motion as a person approaches—slowing, pausing, or stopping to preserve a protective distance [41, 16]. Realizing these policies in motion requires casting clearance and interaction requirements as state and input constraints on position, velocity, and effort; coupling those constraints to what the perception system can deliver reliably; and enforcing them in the low-level loop via calibrated thresholds, hysteresis, and dwell

times so that behavior at the boundary of the protective zone is stable, repeatable, and auditable. We adopt SSM semantics with hysteresis and dwell—distinct STOP/RELEASE bands and a minimum out-of-risk time—to eliminate chatter and make boundary behavior auditable.

All experiments use a synchronized MATLAB–CoppeliaSim loop that shares one clock for sensing, control, and actuation, enabling reproducible STOP/RELEASE events and null-space actions (see §5.1). Units and frame conventions used throughout (world frame W, meters, radians, and per-second rates) are declared once in §3.5 and reused verbatim in Chapters 4–6.

Redundant manipulators are particularly well suited to this setting because multiple joint configurations can realize the same end-effector pose. Task-priority control formalizes the separation between a primary end-effector objective and secondary objectives confined to the Jacobian's null space [1, 2]. Within that null space, posture can be organized, joint limits respected, and collision-avoidance biases introduced without corrupting the commanded task motion. Because redundancy is often exploited near singularities and workspace boundaries, the inverse-kinematics computation must remain numerically well-conditioned; damping the least-squares solution and filtering ill-conditioned directions with singular-value decomposition provide predictable responses while preserving reactivity.

Time parameterization shapes both human interpretability and actuator demand. Trajectories with bounded acceleration and jerk are easier for collaborators to anticipate and impose less mechanical stress. Linear segments with parabolic blends offer closed-form profiles with well-understood transients and straightforward saturation handling, making them practical for real-time tracking and for pause/resume under supervisory control. Around such references, light Cartesian damping and carefully chosen velocity caps suppress residual errors and prevent overshoot when targets are near or when perception updates are intermittent.

1.1 Aim and Motivation

This dissertation introduces methodologies for responsive, collision-aware collaborative manipulation with a redundant robot operating near a person. The overarching aim is to translate high-level safety intent into executable motion: behavior should remain legible to an observer, numerically well-conditioned in the controller, and consistent at proximity thresholds. The proposed architecture integrates a robust inverse-kinematics layer, smooth time-parameterized references, proximity-aware behaviors confined to the robot's redundant degrees of freedom and a compact supervisory logic that governs approach, pause, stop, and recovery in a predictable way. Figure 1.1 summarizes the resulting architecture: bounded-jerk LSPB references, SVD-regularized DLS IK, strict null-space containment, and explicit STOP–HOLD–RELEASE supervision.

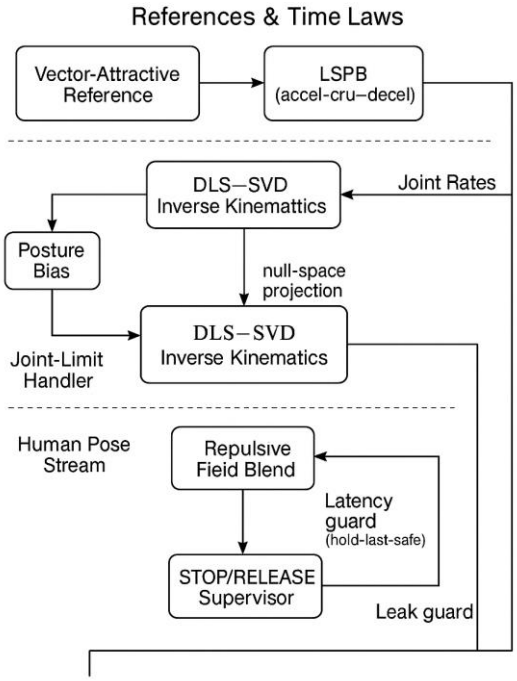

Fig. 1.1 Control architecture adopted in this work. Top: task-space reference generation: vector-attractive (interactive) or LSPB (time-parameterized) with acceleration/velocity bounds. Middle: SVD-regularized DLS IK with posture shaping and joint-limit handling. Bottom: human-aware supervision—skeleton-to-capsule distances, null-space repulsion, and explicit STOP–HOLD–RELEASE gating with leak/latency guards.

Human pose estimates are converted into arrangements of simple geometric volumes aligned with major limbs; distances between these volumes and link-level robot proxies provide the proximity signals that drive both local avoidance tendencies and supervisory gates.

A key part of the work is the alignment of CoppeliaSim and MATLAB into a single, faithful representation of the collaborative cell. Kinematic and dynamic parameters, coordinate frames (including the tool center point), unit conventions, and time-stepping are synchronized so that what is commanded in MATLAB is exactly what executes in the simulator, and what is measured in the simulator is what the controller expects. Communication and logging are organized to preserve timing (controller tick versus physics step), making the virtual cell a realistic stand-in for a physical setup and a reliable platform for repeatable experiments and diagnostics.

Across all scenarios, the implementation relies on a small set of core routines and software modules. A kinematic Jacobian routine provides the geometric Jacobian and related quantities used to convert task-space references into joint commands while keeping numerical conditioning under control. A proximity and collision-avoidance module processes human–robot distances, shapes avoidance tendencies with smooth onsets and caps, and confines these actions to redundant directions so the primary objective is not disturbed. Together with posture and joint-limit management and the supervisory logic, these components form a compact, reusable toolkit.

This approach is validated through five scenarios of increasing complexity. First, a foundational tracking case establishes a clean baseline by driving the tool toward a target with a purely attractive task-space velocity field and no person present, avoiding time parameterization. Second, an interactive extension introduces a nearby operator: the robot advances, then on intrusion halts, holds, gently reshapes posture to increase clearance, and resumes once conditions are comfortable again. Third, a time-parameterized case adopts linear-segment-with-parabolic-blend references to demonstrate smooth, bounded-jerk tracking in the absence of interference. Fourth, a supervised pause–resume variant layers proximity governance onto those references, pausing within a caution band and resuming from a consistent state when the band clears to yield predictable behavior at thresholds.

Finally, a fixed-pose reconfiguration case holds the tool pose constant and exploits redundancy to adjust posture and enlarge human–robot clearance without inducing tool drift, isolating the clearance-management behavior when the primary objective is immovable.

All scenarios are implemented and exercised in CoppeliaSim with MATLAB-driven control and logging. Common health criteria are enforced: no contacts, a minimum clearance margin, and joint-range compliance and performance is reported through task-error histories, minimum-distance timelines, state-transition histories in the supervisor, joint-speed usage, and indicators of numerical conditioning and manipulability. Particular attention is paid to legibility (how the motion reads to an observer), repeatability (how behaviors trigger with thresholds, hysteresis, and dwell), and practicality (how the stack behaves when perception updates are intermittent or when the robot nears kinematic limits).

The contribution to the field is twofold. First, this work offers a unified, implementation-level control stack that maintains tool-level objectives while managing human–robot clearance through redundancy, with behaviors that are transparent to operators and auditors. By combining robust inverse kinematics, either a linear attractive velocity field or time-parameterized LSPB references, a proximity-aware posture-reshaping mechanism, and a lightweight supervisor into a coherent whole, it provides a practical template for responsive collaborative cells. Second, it contributes a reproducible methodology and testbed: a CoppeliaSim-based pipeline that links perception to geometric modelling, supervision, and control, together with diagnostics that expose proximity, effort, and conditioning over time. This combination supports comparative studies and offers a clear route to adapting the approach to other redundant manipulators and sensing suites.

1.2 State of the Art

Research on collaborative manipulation in shared workspaces has converged on a control-centric view in which redundancy and null-space projection are the primary instruments for maintaining human–robot clearance while pursuing task objectives. Classical robot control provides the theoretical backbone: task–priority schemes separate a primary end-effector task from secondary behaviors confined to the

Jacobian null space, allowing posture regulation, joint-limit avoidance, and collision-avoidance postures to coexist with the commanded tool motion [1, 2]. Inverse kinematics is typically regularised through damped least squares with SVD to ensure numerical stability near singularities and workspace boundaries, a practice now standard in redundant manipulation [1, 2, 10]. Recent contributions refine how null-space behaviours are shaped specifically for safe human–robot collaboration: compliance or avoidance fields are injected in the null space so clearance improves without corrupting the primary task, with tunable trade-offs between tracking performance and conservativeness [5, 6, 7, 8].

Within this frame, redundancy is not only a means to avoid singularities but a resource for safety. Analytical parameterizations of 7-DoF arms clarify the redundancy manifold of common cobots and how secondary objectives can be scheduled along it without inducing wrist flips or joint saturation [12, 2]. Surveys on inverse kinematics and control emphasize the practicalities of task-priority control under constraints—damping selection, conditioning metrics, saturation handling, and priority conflicts—which are essential when safety-oriented behaviors run concurrently with tracking [10, 1]. In parallel, human-robot-interaction (HRI)-focused texts argue for legible, predictable motion and transparent supervisory logic, aligning safety behavior with human expectations in shared spaces [4].

Trajectory time-parameterization and legibility are recurring themes. Simple LSPB profiles remain widely used because they bound acceleration and jerk, yield deterministic transients, and pair well with velocity/acceleration caps and pause/resume logic properties valued in human-robot-collaboration (HRC) where humans infer intent from motion [3, 2]. When combined with null-space projection, such profiles allow the end-effector to follow smooth references while the posture adapts in the background to maintain comfortable spacing.

Safety supervision in collaborative cells is commonly organized around speed-and-separation monitoring (SSM). Rather than treating avoidance purely as a potential-field overlay, SSM-oriented designs employ explicit operating modes: approach, caution, pause, stop, recover; with hysteresis and dwell times to prevent chattering at thresholds and to make resume behavior reproducible [4, 8, 7]. In this view, the

supervisor arbitrates between the primary task and safety-motivated null-space behaviors: when proximity becomes critical, progression halts cleanly; when conditions improve, motion resumes from a consistent state.

Perception and proximity modelling underpin these decisions but need not dominate the architecture. A common practical strategy is to reduce human pose data to lightweight geometric abstractions; simple volumes aligned with major limbs, and to approximate robot links with equally simple proxies; these yield fast, smooth minimum-distance queries suitable for control-rate use without committing to a specific sensor brand or modality [1, 2]. Vision-based HRC studies demonstrate that such geometric modelling supports responsive controllers and SSM supervisors across a variety of sensing stacks; examples range from skeleton-based pipelines to multi-view fusion and point-cloud integration, primarily as enablers for the control and supervision layers rather than ends in themselves [11, 16]. Beyond vision, model-based distance surrogates (e.g., signed-distance networks or composite signed-distance-fields (SDFs) for articulated robots) have been explored to accelerate collision queries while preserving controller-friendly gradients, further decoupling the control design from raw sensing idiosyncrasies [9].

From an implementation standpoint, recent work stresses “system transparency”: conditioning measures (singular values, condition numbers), manipulability indices, and saturation flags help diagnose priority conflicts between tracking and avoidance and make safety behaviour auditable [10, 7]. Simulation-in-the-loop workflow commonly combining CoppeliaSim for scene dynamics with MATLAB for control/support rapid iteration and controlled evaluation of state machines, null-space behaviours, and time-parameterised tracking before hardware trials [16]. This tooling aligns with the methodological emphasis in the present work: control-first design, redundancy-aware safety behaviours, explicit supervision, and observability.

Against this background, current work adopts a task-priority architecture with SVD-regularised DLS IK, smooth LSPB references for legibility, and safety behaviours confined to the null space to preserve tool-level objectives [1, 2, 3, 10]. It follows recent HRC trends that modulate posture rather than tool motion whenever possible [5, 6] and employs a compact SSM-oriented supervisor to ensure

predictable approach–pause–resume dynamics [4, 7, 8]. Human pose is mapped to simple volumetric models to obtain controller-rate distance signals independent of any single sensing modality [16, 11, 9]. The emphasis throughout is on implementation-level consistency, conditioning, timing, and repeatability, so that behaviors are both interpretable to users and defensible to auditors.

1.3 Collaborative Robotics in Shared Workspaces

Human–robot collaboration is increasingly framed as the integration of robots into human activities so that people, robots, and the workstation environment operate as a tightly coupled system. Collaboration is not confined to occasional contact; it involves shared commitments in time and space on the same artefacts and coordinated behavior that combines robotic precision and repeatability with human adaptability and judgment. Within industrial settings this has driven a shift away from physical segregation toward cells designed to remove barriers while retaining safety. Modern collaborative arms combine passive features such as lightweight structures and rounded edges with active functions that detect undesired interaction and stop motion when predefined thresholds are exceeded. The aim is a synergistic workspace in which the robot's endurance and accuracy complement human dexterity and cognition, enabling tasks of greater variability and complexity than either agent could manage alone.

Collaboration in practice is organized along a spectrum that couples spatial and temporal sharing:

- **Cell:** the robot operates behind guards; no co-presence.
- **Coexistence:** barriers are removed but human and robot do not work on the same task simultaneously.
- **Sequential collaboration**: human and robot alternate operations at the same station.
- **Cooperation:** both act on the same artefact with limited coupling.
- **Responsive collaboration:** both are in motion on the same artefact and the robot adapts online to human actions.

Current deployments still cluster around coexistence and sequential collaboration because they are easier to certify and operate. The mode that most realizes the potential of HRC, responsive collaboration, demands that the system refresh its understanding of the scene at a rate compatible with control, adjust motion online, and communicate intent through legible kinematics.

Safety in shared spaces rests on two complementary layers. Power-and-Force Limiting bounds the energy exchanged in any incidental contact via risk assessment and limits on forces, torques, speeds, or momentum. Speed-and-Separation Monitoring pre-empts contact by regulating motion as a person approaches; slowing, pausing, or stopping according to clearly defined distance bands. In everyday operation SSM governs behavior; it is implemented through explicit operating modes with thresholds, hysteresis, and dwell times so behavior near boundaries is stable, repeatable, and predictable to non-experts.

Control and kinematics determine how these behaviors are realized. Redundant manipulators, typical of human-scale cobots, admit families of joint configurations for the same tool pose. Task-priority formulations exploit this by separating the tool-center objective from secondary behaviors confined to directions that do not affect the task (the Jacobian null space). Within those directions the robot regulates posture, honors joint ranges, and biases itself away from hazards without corrupting commanded motion. Because collaborative layouts often push arms toward kinematic boundaries, inverse kinematics is commonly regularized, most often via damped least squares with SVD, to maintain numerical stability and preserve predictable responses. At the trajectory level, time-parameterized profiles with bounded acceleration and jerk support legibility, straightforward saturation handling, and clean pause/resume semantics.

Sensing and environment modelling close the loop from intention to action. What the controller needs are timely, numerically well-behaved proximity cues rather than a specific sensing brand. A practical strategy abstracts the operator's body with simple geometric volumes aligned to major limbs and approximates robot links with lightweight proxies; minimum distances between these shapes are then evaluated at the controller update rate and provided to both the supervisor and the

motion generator. This geometry-first approach keeps the design adaptable to different sensing suites and workstation layouts.

1.3.1 Functions and representative uses in everyday settings

Collaborative robotics is not a single application but a family of functions that recur across sectors. Typical functions include:

- **Assisted positioning and fixturing:** the robot holds or pre-positions a workpiece while a person aligns, inspects, or fastens. Examples include door or panel alignment in assembly lines, jigless drilling, and manual fastening on parts that vary slightly batch-to-batch.
- **Co-manipulation and load sharing:** human and robot jointly carry, orient, or insert large or flexible components such as cables, trim, or composite skins, reducing ergonomic strain while preserving human judgment during fit-up.
- **Tool sharing and process assistance:** the robot performs repeatable sub-tasks; screwdriving, sealing, adhesive dispensing, sanding/polishing, while a person handles preparation and quality checks; in craft or repair settings, the robot acts as a third hand for clamping or steadying.
- **Kitting, sorting, and small-batch handling:** collaborative pick-and-place for order preparation, packaging, and co-packing where product mixes change frequently and human oversight resolves ambiguities.

We target shared-workspace tasks where motion must communicate intent and preserve protective distances. In these settings, PFL and SSM are complementary: PFL limits contact severity; SSM regulates approach and halts/resumes with verifiable dwell logic.

- **Human-guided automation:** operators teach new paths by demonstration, then the robot repeats them with higher repeatability; this is common in small and medium enterprises where changeovers are frequent.
- **Laboratory and clinical support:** sample handling, pipetting, or instrument positioning next to technicians; bedside assistance that positions tools or cameras under human supervision.

- **Service and retail demonstrations**: coffee preparation, bar tending, or interactive kiosks where the robot performs structured motions while people operate nearby, highlighting legibility and safety cues.

These uses share practical characteristics: the robot contributes precision, endurance, and repeatability; the human contributes perception, dexterity, and context awareness. Collaboration succeeds when motion communicates intent clearly, when pauses and resumptions are predictable, and when the system returns promptly to productive operation after a cautionary state.

Collaborative workstations are ultimately socio-technical systems. Task allocation (who grasps, who positions, who inspects), layout (reach envelopes, line of sight, escape paths), and communication cues (lights, sounds, on-screen prompts, and the "feel" of the motion) determine whether collaboration is natural and trusted. Transparent instrumentation, conditioning and manipulability indicators, saturation flags, proximity timelines, supports tuning and auditing, while simulation that mirrors the intended physical cell enables safe rehearsal of procedures and systematic evaluation of edge cases before human involvement.

1.4 Work Description

The work reported in this thesis builds a practical pathway from high-level safety intent to executable motion for collaborative manipulation with a redundant arm. The overarching idea is to keep task behavior legible, to use redundancy for conservative postural adjustments near people, and to make slow/hold/resume decisions transparent and reproducible. To ground this idea in concrete, verifiable artefacts, the chapter moves from system context to implementation and evidence: it first establishes the experimental platform, then explains how human motion is represented and consumed by the controller, and finally presents the staged controller configurations and the criteria used to evaluate them.

1.4.1 Experimental platform

A 7-DoF Franka Emika Panda operates in CoppeliaSim while kinematics, supervision, and control execute in MATLAB. Figure 1.2 shows the 7-DoF Franka platform used in our experiments.

Fig. 1.2 Franka Emika Panda/FR3 collaborative arm used as the primary manipulator in this work.

The simulator runs in synchronous mode: each control tick advances physics by one fixed step. The controller period is an integer multiple of that step so reference sampling, Jacobian evaluation, and supervisory transitions share the same clock. Frames (base, flange, TCP) are matched across tools; the TCP is verified by forward–inverse round-trip checks; joint ordering and limits are cross-checked against the simulator model. Lightweight geometric proxies are attached to links for distance queries. All logs (poses, joint states, distances, modes) carry control-tick timestamps for lossless alignment

1.4.2 Human motion: Modelling and MATLAB implementation

Human motion is ingested as time-stamped 3D skeletons (major joints). Poses are normalized by anchoring a torso frame, aligning axes to the simulator convention, reconciling units, interpolating short gaps, and low-pass filtering to suppress jitter while preserving natural limb swings. Retargeting maps joint pairs (shoulder–elbow, elbow–wrist, …) to limb segments; per-sequence segment lengths keep

proportions consistent across operators. Each segment becomes a capsule/sphere with conservative radii; robot links are approximated by aligned proxies. At each tick we compute minimum distances over configured human–robot pairs, debounce them, and rate-limit changes to enforce physically plausible approaches. Conditioned distances feed both the supervisor (mode gating) and the null-space postural shaper (repulsion that preserves the tool objective).

1.4.3 Core software components

The implementation relies on a small set of named, reusable elements: a kinematic Jacobian routine returning the geometric Jacobian and conditioning indicators each tick; a posture-bias routine that converts the de-bounced distance vector into smooth, bounded postural references when task progression is permitted; a gated-avoidance routine that strengthens avoidance and suspends task-space commands when risk bands are exceeded; and human-model utilities that provide pose ingestion, retargeting, volume instantiation, and animation/replay for repeatable experiments.

1.4.4 Control architecture

Task execution follows a task-priority formulation with a numerically regularized inverse-kinematics layer. Section 3.4 details the two TCP time-laws (vector vs. LSPB) and their pause/resume semantics; Section 3.5 fixes the safety thresholds used throughout. Tool-center reference, either Cartesian velocities or time-parameterized trajectories, are mapped to joint commands by a damped least-squares solver (SVD) so responses remain well conditioned near singularities and joint limits. Posture regulation, joint-limit avoidance, and proximity-aware biases act strictly in the Jacobian null space so the commanded tool motion remains intact whenever redundancy allows. Orientation locking near the target prevents wrist flips; an acceptance radius and a terminal-speed floor make arrivals reproducible. When time parameterization is required, trajectories follow linear-segment-with-parabolic-blend profiles; sampling is tied to the physics step for discrete-time consistency, and modest Cartesian damping with conservative velocity caps suppresses residual errors and overshoot. A compact supervisor implements operating modes—track, caution, pause, stop, recovery—with calibrated

thresholds, hysteresis, and dwell times to avoid chattering and to guarantee predictable resumption.

1.4.5 Staged experiments

Behavior is probed in five configurations of increasing richness. The sequence begins with clean target acquisition using a purely attractive Cartesian velocity field in the absence of a person, establishing baseline tracking and conditioning. Next, a nearby operator is introduced: as proximity tightens the controller halts and holds, reshapes posture through redundancy to enlarge clearance, and resumes smoothly once conditions are comfortable again. The third configuration replaces the attractive field with LSPB trajectories to demonstrate smooth, bounded-jerk tracking and straightforward saturation handling. The fourth applies proximity governance to those trajectories so the path pauses deterministically within a caution band and resumes from a consistent state when the band clears, with null-space posture shaping active throughout. The final configuration fixes the tool pose and asks the arm to reconfigure through redundancy alone to increase human–robot clearance, isolating posture control and checking for negligible tool drift.

1.4.6 Data flow, logging and evaluation

Each control tick reads joint state and tool pose, ingests the de-bounced distance vector, and queries for the Jacobian and conditioning indicators. A task-space command (velocity or LSPB sample) is formed; the output of the posture-bias or gated-avoidance routine is projected into the null space; task and redundancy components are summed, capped, and sent to the simulator. Projection and smoothing precede integration to preserve discrete-time correctness. The logger records joint states, tool poses, distances, supervisor mode and transition causes, singular values, manipulability, saturation flags, bias magnitudes, and command histories. Performance is assessed on safety/feasibility (no contacts; minimum clearance margin; joint limits respected), tracking quality (rms/peak tool error; acceptance-radius and terminal-speed behavior; no overshoot in time-parameterized runs; residual drift while holding), proximity management (minimum-distance timelines; time in each mode; pause/resume counts and durations; clearance growth while fixed), numerical health (smallest singular value,

condition number, manipulability; time in saturation; velocity/acceleration usage; alignment between tracking and avoidance), and legibility/repeatability (smooth transients; consistent thresholds/hysteresis; reproducible mode transitions under replayed human motion). Experiments are repeated from varied initial postures and re-playable human traces; ablations disable specific elements (e.g., null-space bias, damping, hysteresis) to isolate their effects.

1.4.7 Contributions

The thesis contributes: (i) a unified control stack that preserves tool-level objectives while managing human–robot clearance strictly through redundancy; (ii) a discrete-time implementation method in which reference sampling is tied to the physics step and projection/smoothing precede integration so resume after pauses is deterministic; (iii) a geometry-first proximity pipeline that converts pose streams to controller-rate distance cues via simple limb-aligned volumes and link proxies, remaining agnostic to sensing brands; (iv) an SSM-oriented supervisor with calibrated thresholds, hysteresis, and dwell integrated with time-parameterized tracking and redundancy-aware posture shaping; (v) a synchronized MATLAB–CoppeliaSim environment and logging scheme that mirror a physical cell; and (vi) a staged evaluation suite with common metrics intended as a template for comparative studies.

1.4.8 Perspectives

The artefacts assembled here are designed to transfer cleanly to hardware-in-the-loop and on-robot trials: the synchronous timing model, controller-rate distance signals, and explicit supervision map directly to real-time middleware. Near-term extensions include substituting live pose sources for recorded streams, incorporating certified reference-governor layers to formalize pause/resume envelopes, and enriching the proximity model to include tools and workpieces. Longer-term, the same architecture can support learned postural priors filtered for safety, multi-arm cells coordinating null-space behaviors, and digital-twin deployments that tie logged indicators (conditioning, manipulability, proximity timelines) to line-level metrics such as cycle time and ergonomic load.

1.5 Thesis organization

This dissertation is structured to move from system-level motivation and style to kinematic methods, human–robot safety mechanisms, implementation, and staged experimental evidence, before closing with a literature-grounded discussion and the concluding outlook.

Chapter 1 – Introduction: The opening chapter states the aim and motivation for safe, legible human–robot collaboration with a 7-DoF Franka Emika Panda in a shared bench-top cell, surveys the state of the art, frames collaborative operation in shared workspaces, and delineates the work description that anchors the remainder of the thesis (Sections 1.1–1.4). These parts set the problem, scope, and contributions that subsequent chapters elaborate.

Chapter 2 – Collaborative Work-Cell Architecture and Safety Framework: This chapter describes the overall system architecture of the collaborative cell, including sensing, supervision, interface contracts, and safety instrumentation. It introduces the components, their dataflow, and the invariants required for deterministic operation; it also outlines transfer/extension perspectives that are revisited after the experiments.

Chapter 3 – Kinematic Model, Time-Law References, and Safety Variables: Here the thesis adopts an operational 6×7 kinematic formulation for the Panda in CoppeliaSim, explains how it is exercised in a synchronized human–robot scene, and fixes the interfaces used throughout (e.g., translational DLS–SVD IK, null-space projector, and supervisor thresholds). The chapter then develops the two TCP reference generators—vector-attractive versus LSPB with pause/resume semantics—and consolidates the global safety variables and thresholds (distance hysteresis, dwell, tracking tolerances, leak bounds) that standardize logging and diagnosis for the experiments.

Chapter 4 – Human Model, Distances, and Safety Behaviors: The HRI layer is formalized: skeleton-derived capsule proxies, clearance distances and nearest-pair queries, and two safety behaviors—continuous repulsive fields blended with the posture bias, and an SSM-style supervisor with explicit STOP/RELEASE

hysteresis and dwell that pauses/resumes an LSPB time law without corrupting its schedule. The chapter then extends to fixed-TCP avoidance in redundancy with leak-bounded null-space action and provides the variables, thresholds, and health flags reused later.

Chapter 5 – Implementation & Software Architecture (CoppeliaSim): This chapter documents the deterministic six-stage per-tick pipeline—from scene input/output (I/O) and geometric lifting, through distance queries and supervisory logic, to task-space tracking and joint-space synthesis—together with post-processing, determinism/test hooks, and the mode scripts that instantiate operating behaviors (5 scenarios). It also details runtime monitors and reproducibility provisions (seeds, artifacts, log bundles).

Chapter 6 – Experimental Evaluation (Scenarios S1–S5): Using the unified MATLAB↔CoppeliaSim stack, five scenarios progressively introduce supervision, human proximity, LSPB timing, and fixed-TCP null-space avoidance. A uniform metric dictionary and logging protocol underpin the figures/tables and the reproducibility checklist. Representative results (e.g., LSPB ramp–cruise–ramp tracking with dwell compliance, low conditioning numbers, strict null-space containment) are reported alongside scenario-specific settings and outcomes.

Chapter 7 – Discussion in the Context of the Literature: The evidence from Chapter 6 is positioned against core HRC themes: strict null-space containment to preserve tool-level objectives, explicit SSM hysteresis/dwell semantics for predictable pause/release, and controller-rate, sensor-agnostic proximity signals (skeleton-to-capsule distances, link proxies). The discussion is organized around the staged scenarios and the synchronized loop that makes timing/conditioning comparable to prior work.

Chapter 8 – Conclusions & Future Work: The thesis closes by distilling contributions and outlining future extensions; these are framed in terms of method generalization, formal safety supervision, richer distance fields, and hardware transferability (as previewed by the architecture and metrics fixed earlier).

Chapter 2

Collaborative Work-Cell Architecture and Safety Framework

This chapter formalizes the collaborative work-cell and safety framework that the rest of the thesis depends on, specifying both the physical stack (7-DoF Franka Emika Panda, sensing suite, calibration artifacts, and fixtures) and the synchronized MATLAB↔CoppeliaSim software loop that drives experiments under deterministic timing. We define global and tool frames, hand–eye and scene calibrations, and the transform registry that guarantees a single source of truth for kinematics and distances; we then make explicit the tick-level contracts—clock source, cycle time, jitter and latency budgets, message ordering, and failure semantics—that bound all controller and supervisor reactions. The cell is organized as a three-layer pipeline: (i) scene I/O and geometric lifting, which ingests raw streams (robot state, human skeleton) and emits rigid-body poses plus capsule proxies with health flags; (ii) proximity and safety signaling, which computes nearest-pair distances, applies hysteresis and dwell timers, and exposes a small, typed interface of safety variables; and (iii) motion generation and supervision, where LSPB time-law references and damped least-squares (SVD) tracking are guarded by an SSM-style supervisor that can STOP and RELEASE without corrupting the LSPB schedule. Throughout, we enforce strict null-space containment for avoidance and posture shaping so that corrective actions do not leak into task-space objectives; we also codify bounds on joint limits, velocity/acceleration, and manipulability to prevent pathological configurations. Finally, we specify logging schemas (signals, units, sampling), determinism hooks (seeds, mode scripts), and integrity checks (range assertions, timeout escalations, safe fallback states), so that Chapters 3–6 can build on a reproducible, auditable, and implementation-ready foundation.

2.1 Shared-workspace scenario and cell layout

This section fixes the geometry and conventions of the collaborative workcell used in all experiments. The setting is a bench-top scene in CoppeliaSim where a 7-DoF Franka Emika Panda and a virtual human share a rectangular table. MATLAB runs the kinematics, supervision, and control; CoppeliaSim provides geometry and physics in synchronous stepping so every control tick advances the scene by one fixed step [16]. The intent is to keep frame definitions and regions explicit so that proximity, gating, and tracking later in the chapter have a precise spatial meaning and can be reproduced.

A single human stands along one long edge of the table and manipulates parts on the surface. The Panda is mounted approximately at the midline of the opposite long edge so the tool center point (TCP) covers the central task zone without pushing joints toward their limits. The nominal task zone is centered on the tabletop, positioned so the TCP works in the robot's dexterous region, well inside joint limits and away from singular postures, with comfortable clearance to the table edges. Targets used in later scenarios lie within this zone and are chosen to avoid posture flips during approach and to keep the tool on the robot half of the bench.

A simple frame hierarchy is used consistently in the simulator, controller, and logs:

- **World frame:** fixed to the table; x runs along the long edge, y points from the human side toward the robot side, and z is vertical.
- **Robot base frame**: rigidly attached to the Panda model; its world→base transform is measured once at scene setup and treated as constant.
- **TCP frame:** attached to the flange; its z-axis is aligned with the nominal approach direction used in the scenarios (downward toward the table). Units and frame conventions used throughout (world frame W; meters, radians, and per-second rates) are declared once in §3.5 and reused verbatim in Chapters 4–6.

For analysis and visualization a torso-anchored human frame is maintained, and limb-aligned segment frames (introduced in §2.3) are used internally when constructing the simple geometric volumes employed for proximity queries. The

human region itself is modelled as a rectangular prism along the far edge of the table; it captures typical bench-top actions such as leaning in to place or remove a part, and brief withdrawals to the edge of the region. Its width spans the fixture area plus a small lateral buffer; its depth allows a natural stance; and its height extends from floor to chest so hands and forearms are represented when the operator leans over the surface. This region is not a guard; it is the reference volume used to position the human surrogate and to define the link–limb distances monitored at control rate.

Clearance reasoning uses lightweight geometry on both sides. Each robot link is paired with a simple proxy volume aligned to its local frame; the tool (when present) has its own proxy. On the human side, limbs are represented by simple volumes aligned to segment axes (§2.3). A fixed set of links–limb pairs is monitored continuously (for example, upper-arm↔upper-arm, forearm↔forearm, hands↔tool) so that minimum-distance queries focus on the interactions that occur at a table rather than wasting computation on irrelevant combinations. A nominal approach vector (the TCP z-axis) is recorded per target so the supervisor can prefer deceleration aligned with the final approach. Keep-out margins at the table's human edge prevent the TCP from overhanging the operator side during automated approaches; these margins are the same ones later used to define "stop" bands in the SSM-inspired policy.

The first implementation step was a mapping layer that guarantees MATLAB and CoppeliaSim represent the same geometry. Homogeneous transforms for world→base and base→TCP are stored in a registry and used identically by the simulator and the controller. The mapping was validated by round-trip checks (pose → inverse kinematics → forward kinematics) and by placing calibration points on the table: points transformed in MATLAB coincide with the same locations in CoppeliaSim within numerical tolerance. This alignment is what allows logs, figures, and controller decisions to have an unambiguous spatial meaning throughout the projects.

For reproducibility, each run logs the world→base and base→TCP transforms, the operator-region dimensions, the list of monitored links–limb pairs, and all target poses with their tolerances. With these metadata, any later plot of minimum

distance, state transitions, or tracking error can be traced to an exact cell geometry and frame convention.

2.2 Robot Model and Link-Proxy Representation

This section formalizes the robot side of the workcell: the kinematics, the controller reasons about, the frame conventions anchoring all geometric quantities and the surrogate link geometry used for real-time clearance evaluation. The Franka Emika Panda model embedded in CoppeliaSim is treated as the single source of truth; MATLAB mirrors its joint ordering and link frames so quantities computed in MATLAB and rendered in CoppeliaSim refer to the same configuration at every control tick.

2.2.1 Ground-truth kinematics

Forward kinematics and the geometric Jacobian are evaluated against the simulator's exact frame definitions. SVD-regularized DLS inverse kinematics, manipulability indices, and link-proxy placement are therefore referenced to the same geometry that drives rendering and collision. With synchronous stepping, forward kinematics/jacobian (FK/J) evaluation, projection, and integration share the simulator's clock, eliminating frame/sign drift and timing skew; edits to tool offsets or base placement are made once in the scene and propagate automatically, improving fidelity and reproducibility [1, 2].

2.2.2 Kinematic description and numerical health

The Panda is a seven-revolute-joint arm with redundancy advantageous for posture shaping. At each tick, the TCP pose is obtained by composing the simulator's transforms; the spatial Jacobian is factorised via SVD to log the smallest singular value, condition number, and a manipulability index. These indicators are later used to interpret slow/hold/resume events as proximity-driven or authority-limited. Orientation uses rotation matrices internally and quaternions in logs to avoid parameterization artefacts.

2.2.3 Frames and transform registry

The world frame is fixed to the tabletop (x along the long edge, y toward the human side, z vertical). Figure 2.1 sketches the world, base, and TCP frames and the transforms used in this work.

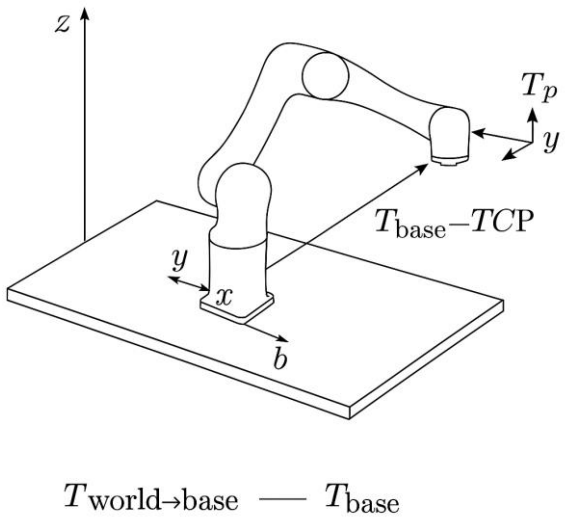

Fig. 2.1 World–base–TCP frames and transform checks. The world frame is fixed to the tabletop; the base frame is attached to the Panda base; the TCP frame is attached to the flange and updated by FK. Transforms are round-trip checked (pose → IK → FK) to verify consistency between MATLAB and CoppeliaSim.

The base frame is rigidly attached to the Panda; its world→base transform is measured once and stored. The TCP frame is attached to the flange and updated by FK. A shared transform registry and round-trip checks (pose → IK → FK) verify concordance between MATLAB and CoppeliaSim within numerical tolerance.

2.2.4 Link-proxy geometry

Exact mesh distances are replaced by conservative primitives (capsules/cylinders for elongated links, spheres for compact ones) rigidly attached to link frames. Parameters are chosen to bound meshes with a small inflation margin. This yields closed-form sphere–sphere/sphere–capsule/capsule–capsule distances that are smooth in time and inexpensive to evaluate; properties essential for high-rate supervision without chatter [17, 18, 21].

2.2.5 Monitored pairs and signals

Only plausible interactions for the bench-top layout are tracked (e.g., proximal links vs. upper arms, mid–distal links vs. forearms, terminal link/tool vs. hands). Table 2.1 enumerates the link–limb pairs monitored at each controller tick and how their signals are consumed.

Link (robot)	Limb (human)	Rationale	Signal used (global/per-pair)
Link 1–2 (proximal)	Upper arm	Likely closest during approach	global + per-pair
Link 3–5 (mid–distal)	Forearm	Mid-reach	per-pair
Link 7 / Tool	Hand	Near manipulation	global + per-pair

Table 2.1 Monitored link–limb pairs and signal usage. For each pair, d_{min} is logged every tick; the supervisor consumes the global minimum for mode gating, while the per-pair vector biases null-space posture to increase spacing from currently critical limb.

Per-pair minimum separations are computed each tick; the supervisor consumes the global minimum for mode gating, while the full vector biases null-space posture to increase spacing from currently critical limbs. Figure 2.2 illustrates how per-pair distance signals generate a posture bias confined to the Jacobian null-space. Distances are debounced and rate-limited before use.

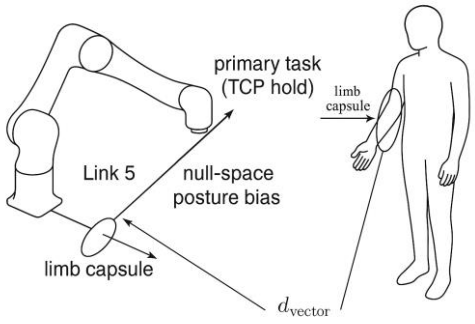

Fig. 2.2 Posture bias in the null-space from per-pair distances. The redundancy policy steers joints away from the currently critical human limb without altering the primary task.

2.2.6 Target-aligned approach

Each target carries a nominal approach direction (TCP z at the goal) used to shape terminal deceleration and holding behaviour and to diagnose conflicts between tracking and avoidance.

Diagnostics and logging. Every tick records world→base and base→TCP transforms, joint states, per-pair and global distances, Jacobian singular values and condition number, manipulability, and joint saturation flags, providing the evidence base for later analyses of proximity management, dexterity, and control effort (see §5.1 for synchronized logging and replay).

2.3 Human pose acquisition and geometric modelling

This section describes how human motion enters the control loop and how it is represented for clearance evaluation. The pipeline has two complementary roles: (i) animate a human mannequin in CoppeliaSim so the scene reflects realistic operator motion; and (ii) produce smooth, control-rate distance signals between the person and the robot's link proxies that the supervisor and redundancy-aware control can consume.

2.3.1 Pose acquisition and normalization

Human motion is provided as a time-stamped skeletal pose stream containing 3D joint key-points for the major limbs. The stream is normalized before use: a torso-anchored reference frame is established, axes are aligned with the workcell's world frame, units are reconciled, short gaps are bridged by interpolation, and jitter is attenuated with a low-pass filter chosen to preserve natural limb swing. A simple retargeting step maps joint key-points to limb segments (e.g., shoulder–elbow, elbow–wrist), with segment lengths estimated per sequence so proportions remain coherent across operators [11, 16].

2.3.2 Mannequin animation in simulation

The normalized pose stream drives a full-body mannequin in CoppeliaSim at the controller update rate, so the virtual operator moves like the recorded one. Each

tick, the mannequin's torso and limb segment frames are updated from the pose stream, yielding a visually faithful representation that also anchors the geometric abstractions used for clearance. This decouples visualization from control: the mannequin conveys what the operator is doing, while separate, lightweight volumes provide the numerically well-behaved distances needed by the controller.

2.3.3 Geometric abstraction for clearance

Clearance reasoning uses elementary volumes aligned to human limbs and robot links. On the human side, each limb segment is instantiated as a simple geometric volume aligned to its segment axis (cylindrical or capsule-like where appropriate; spherical for compact parts such as hands). On the robot side, link-aligned proxies are defined as in §2.2. Minimum distances are then evaluated between a fixed set of limbs–link pairs (e.g., upper arm vs. proximal links, forearm vs. mid–distal links, hands vs. terminal link/tool), chosen to reflect plausible interactions at a bench-top station.

In addition to limb volumes, a torso-centred keep-out cylinder (diameter ≈ 0.40 m) defines a conservative personal space around the operator. While primarily a visual and supervisory aid, it yields a single, intuitive scalar; the tool-to-torso-zone distance that complements the per-pair limb distances and provides a coarse warning band for approach/hold decisions.

These abstractions admit closed-form distance queries (sphere–sphere, sphere–cylinder/capsule), which are smooth under motion and inexpensive to compute, ensuring that proximity can be evaluated every control tick without numerical artefacts [9, 17, 18, 21].

2.3.4 Distance computation and signal conditioning

For each monitored limb–link pair, the minimum separation is computed at the controller tick. Two signals are produced: (i) the global minimum across all pairs, used by the supervisor to escalate modes (track → caution → pause/stop → recovery), and (ii) the full vector of per-pair distances, used by the redundancy policy to bias posture away from whichever limb currently dominates proximity.

Before entering the control stack, distances pass through a short-horizon debouncer and a rate limiter so that isolated spikes or unrealistically fast changes do not provoke chattering or oscillatory mode switching.

2.3.5 Interfaces and timing

The entire pose-to-distance pipeline is clocked by the same synchronous stepping used for robot control: one controller tick updates the mannequin, regenerates limb volumes, evaluates all limb–link distances, conditions the signals, and publishes the global minimum and per-pair vector to both the supervisor and the posture-shaping process. Figure 2.3 outlines the synchronous pose-to-distance pipeline executed each controller tick.

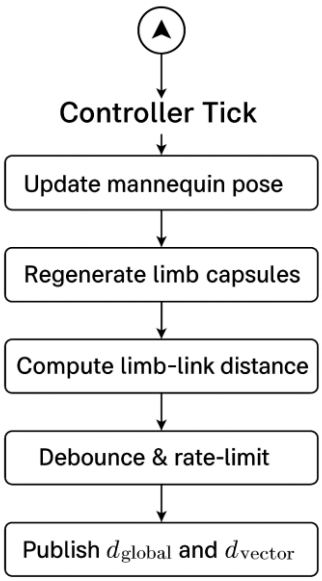

Fig. 2.3 Synchronous pose-to-distance pipeline. Each tick: mannequin update → limb-volume generation → limb–link distance evaluation → debouncing and rate limiting → publication of global minimum and per-pair vector to the supervisor and the null-space posture shaper.

This ensures that animation, proximity, and control share a common time base and frame convention, eliminating hidden latencies between what is seen in the scene and what the controller reacts to.

2.3.6 Reproducibility

Each run stores the pose-stream identifier and sampling rate, the limb-volume parameters (segment radii and lengths), the list of monitored limbs–link pairs, and the keep-out cylinder dimensions, alongside the logged distance timelines. With these metadata, distance plots and state-transition histories elsewhere in the thesis can be traced back to an exact human-model configuration and timing.

2.4 Proximity metrics and clearance policy

This section specifies how proximity is quantified and how those quantities govern motion. The objective is a policy that is numerically well-behaved at control rate, transparent to audit, and predictable to an observer: slow early, hold decisively, and resume smoothly once comfortable spacing is re-established.

2.4.1 Distance signals and conditioning

At each control tick the workcell computes minimum separations between a fixed set of limb-aligned human volumes and link-aligned robot proxies (defined in §2.2–§2.3). Two signals are produced:

- A global minimum distance, used to gate operating modes;
- A vector of per-pair distances, used to bias posture in redundant directions.

Before entering the supervisor and the posture shaper, distances pass through a short debouncing filter and a rate limiter. Debouncing removes isolated spikes (e.g., transient pose jitter); rate limiting enforces physically plausible approach speeds so that supervisory logic is not driven by artefacts. Conditioning is strictly causal and bounded so that latency is predictable and small relative to the control period.

2.4.2 Clearance bands and invariants

Clearance is organized into three concentric bands around the robot–human separation:

- **safe band:** normal tracking is permitted;

- **caution band:** tracking is allowed but conservative behaviour is encouraged (reduced speeds, stronger postural bias away from the nearest limb);
- **stop band:** motion toward the goal is suspended; only posture reshaping in redundancy (and any necessary damping) remains active to increase spacing.

Band limits are chosen with respect to the bench-top layout (table depth, reach envelopes) and are expressed in the world frame so they are invariant to the robot's posture. Two invariants govern behaviour:

- **Monotonic escalation:** once a more conservative mode is entered (safe → caution → stop), the system cannot jump directly to a less conservative mode without first satisfying the exit conditions of the current one;
- **Non-chattering transitions**: all band boundaries are paired with hysteresis margins and minimum dwell times so that brief fluctuations do not cause oscillatory mode switching.

The caution band is typically paired with a gentle reduction of commanded tool speed and an increase in postural bias; the stop band enforces a hold at the current task progress while redundancy is used to expand clearance.

2.4.3 Mapping distance to postural demand

The per-pair distance vector is converted to a "clearance demand" that shapes posture inside the Jacobian null space. The mapping obeys three principles:

- **smooth onset:** demand rises continuously as a limb approaches the caution band, avoiding discontinuities in joint commands;
- **saturation**: demand caps at a finite level to prevent excessive joint velocities even when a limb is very close;
- **locality:** only the pairs currently near their limits contribute materially, so posture changes are relevant to the active interaction.

This demand does not interfere with the primary task directions; it is confined to redundant directions so that tool motion proceeds unchanged whenever redundancy

permits. When redundancy is exhausted (e.g., near singularities or joint limits), the supervisor, not the postural shaper, resolves the conflict by reducing or suspending task progression.

2.4.4 Gate logic and timers

Mode transitions are driven by the global minimum distance, subject to hysteresis and dwell:

- enter caution when the global minimum falls below the caution threshold; exit when it rises above the caution threshold plus hysteresis for at least the dwell time;
- enter stop when the global minimum falls below the stop threshold; exit when it rises above the stop threshold plus hysteresis and remains there for the dwell time.

While stopped, the controller maintains a stable hold: task references are frozen; posture reshaping continues; damping and velocity caps remain active. Resumption re-enables tracking from the frozen reference (or, for time-parameterised runs, from a consistent resume point) so that motion continues without discontinuities.

2.4.5 Terminal behaviour at targets

To make arrivals legible and repeatable, target definitions (see §2.1) include an acceptance radius, an orientation tolerance, and a nominal approach direction (TCP z at the goal). Near the goal the supervisor enforces a terminal-speed floor and an orientation lock to avoid wrist flips. If a proximity event occurs inside the acceptance radius, the hold is performed with respect to the recorded approach direction; resumption continues along that direction to the same terminal pose.

2.4.6 Consistency and logging

All policy decisions are tied to the control clock used for animation and kinematics. Every tick records the global minimum distance, the per-pair vector, the current mode, any transition event (with reasons and timestamps), and the instantaneous values of the thresholds, hysteresis margins, and dwell timers. This record allows

later chapters to trace slow/hold/resume decisions to measurable proximity conditions and to verify that the invariants (monotonic escalation, non-chattering transitions) were respected.

2.4.7 Scope of the envelope

The clearance policy guarantees: (i) no commanded progression toward the goal while within the stop band; (ii) conservative tracking within the caution band with redundancy-confined posture reshaping; and (iii) deterministic resume from a well-defined state once clearance persists beyond the release thresholds. It does not, by itself, certify contact forces; rather, it provides the proactive separation management on which the rest of the control architecture builds.

2.5 Supervisory gating and operating modes

This section formalizes the state machine that governs approach, slowdown, holds, and resumptions in the shared workspace. The supervisor sits between proximity signals (§2.4) and motion generation (§2.2–§2.3), producing at each control tick a small set of directives: whether task progression is permitted, the current speed scale for tool motion, and the gain schedule for redundancy-confined postural reshaping. Its design targets three properties: monotonic escalation (once behaviour becomes more conservative it cannot immediately become less so), non-chattering transitions (thresholds are paired with hysteresis and dwell times), and deterministic resume (motion continues from a well-defined, reproducible state).

2.5.1 Mode set and responsibilities

The supervisor operates over a finite set of modes:

- **init:** one-time alignment and health checks after start/reset.
- **track:** normal task execution; tool motion follows the reference; postural reshaping is present but minimal.
- **caution:** task execution continues with a conservative speed scale; postural reshaping gains increase to bias the arm away from the nearest limb(s).

- **hold:** commanded progression toward the goal is suspended; only damping and redundancy-confined postural reshaping remain active to enlarge spacing.
- **recovery:** a short, deterministic ramp that re-enables task motion after a cleared hold; speed and postural gains return smoothly to track values.
- **fault (latent):** entered on stale/invalid proximity data or internal consistency violations; the system behaves like a hold until data integrity is restored, then proceeds through recovery.

Each mode emits a tuple (task_enabled, speed_scale, posture_gain) and a small set of flags (orientation-lock, terminal-speed floor). The orientation lock prevents wrist flips near the goal; the terminal-speed floor ensures legible arrivals.

2.5.2 Transitions, thresholds, and timers

Mode transitions are driven by the conditioned global minimum distance d_{min}(from §2.4), with distinct entry and release thresholds to realise hysteresis:

- track → caution when $d_{min} < d_{caught}$. Release to track when $d_{min} > d_{caught} + h_{caught}$ for at least T_{caught} .
- caution → hold when $d_{min} < d_{stop}$. Release to recovery when $d_{min} > d_{stop} + h_{stop}$ for at least T_{stop} .
- recovery → track after a fixed ramp time T_{rec} or once the commanded speed scale reaches 1 with bounded jerk.

Thresholds satisfy $d_{stop} < d_{caught}$ and margins h_{stop}, $h_{caught} > 0$. Dwell times T_{stop} and T_{caught} are chosen as small integers of the control period to keep timing discrete and auditable. The fault mode preempts all others: it is entered if proximity data are stale beyond T_{stale} if distances become non-finite, or if internal consistency checks (e.g., contradictory timers) fail.

2.5.3 Actions per mode

Figure 2.4 summarizes the supervisor's actions in each mode:

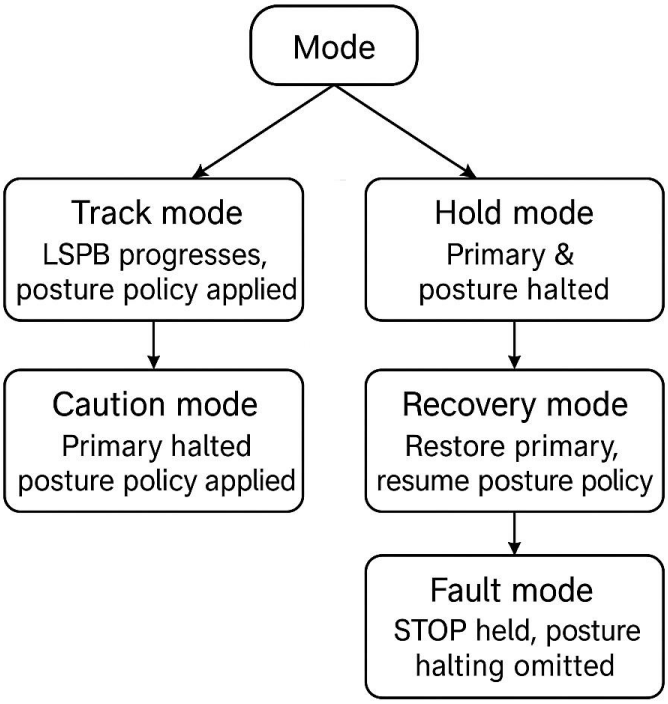

Fig. 2.4 Supervisor actions per mode. Distances gate transitions; within each mode the supervisor sets (task_enabled, speed_scale, posture_gain) and applies orientation lock or terminal-speed floor near targets. Recovery uses a jerk-limited ramp; fault mirrors hold and requires restored data integrity before recovery.

- **track:** task progression enabled; speed_scale = 1; posture_gain at baseline; velocity caps and light Cartesian damping active.
- **caution**: task progression enabled; speed_scale reduced via a smooth, distance-dependent map; posture_gain increased with smooth onset; velocity caps tightened.
- **hold:** task progression disabled; the last valid tool reference is frozen (Cartesian-velocity case) or the time-parameterised reference index is held (trajectory case). Damping remains; posture reshaping continues in the Jacobian null space to enlarge clearance; the commanded tool velocity along the recorded approach direction is zero.
- **recovery:** task progression re-enabled with a jerk-limited ramp of speed_scale from 0→1; posture_gain decays to baseline; if a time-parameterised trajectory is used, resumption occurs from the frozen index (or a re-timed, nearby sample) to avoid discontinuities; otherwise the Cartesian-velocity generator is warm-started from the held pose.

- **fault:** identical to hold with an additional requirement that data integrity be restored for T_{stale} before entering recovery.

2.5.4 Interaction with references and IK

The supervisor never edits the reference's geometry; it gates access to it. In Cartesian-velocity operation, gating sets the commanded tool twist to zero while preserving the integrator state; in time-parameterised operation (e.g., LSPB), gating freezes the reference sample index and resumes without skipping, so the same geometric path is followed with inserted dwell. In all modes, redundancy-confined postural reshaping is computed before projection/integration on the same tick (the "compute → project → smooth → integrate" ordering), ensuring discrete-time correctness and preventing null-space actions from leaking into primary task motion.

The inverse-kinematics layer (SVD-regularised DLS) runs at every tick regardless of mode, but its input is modulated: in hold/fault, only the null-space component and damping remain; in caution, both task and null-space components are present but scaled. Near the goal, the orientation lock fixes the tool's rotational setpoint to avoid wrist inversions; simultaneously, a terminal-speed floor prevents the commanded speed from asymptotically vanishing, yielding decisive "arrive and stop" behaviour.

2.5.5 Use of approach direction and keep-out margins

Each target carries a nominal approach direction (TCP z at the goal) and keep-out margins at the table's human edge (§2.1). In caution/hold, deceleration and holding are resolved with respect to this direction: vertical slow-downs near the surface and holds that do not creep laterally across the table edge. Keep-out margins align the stop threshold with workspace geometry, ensuring that automated approaches do not overhang the operator side.

2.5.6 Priority, concurrency, and edge cases

- **Priority:** fault > hold > caution > track. Recovery only follows a cleared hold/fault.

- **Concurrency:** if dexterity degrades (e.g., smallest singular value below a limit) while in caution, the supervisor may tighten speed caps or escalate to hold even if d_{min} has not crossed the stop threshold, preventing large joint excursions during near-singular operation.
- **Goal inside a hold:** if the target is reached (within acceptance radius and orientation tolerance) while held, the system records completion but remains in hold until release conditions are met; on recovery it transitions directly to track-idle (no further motion).
- **Lost target:** if target validity is withdrawn (e.g., upstream task reset), the supervisor enters fault, freezes motion, and awaits a consistent target before recovery.
- **Stale distances during recovery:** if proximity becomes stale during the ramp, recovery is aborted and the system returns to fault/hold.

2.5.7 Parameters and tuning guidelines

Thresholds d_{caught}, d_{stop} are set relative to the monitored limb–link pairs most likely to dominate around the table (hands vs. terminal link/tool typically define d_{stop}). Hysteresis margins are at least one to two ticks' worth of the maximum plausible distance change (from the rate limiter), ensuring non-overlap. Dwell times are chosen to exceed the longest filter horizon in the proximity pipeline, guaranteeing that transitions are driven by sustained conditions rather than filter transients. Speed-scale maps are monotone with bounded slope to keep commanded accelerations within actuator limits during recovery.

2.5.8 Timing and logging

All decisions are tied to the synchronous control clock; timers advance by whole ticks, and transition guards evaluate the condition plus elapsed dwell at the tick boundary. Every event (entry/exit with reason, thresholds in effect, dwell counters) is logged alongside d_{min} ,the per-pair distance vector, mode, speed_scale, posture_gain, and IK conditioning statistics. This record enables audit of each slow/hold/resume and supports replication of runs with identical outcomes.

2.5.9 Guarantees

Given valid proximity signals and the configured thresholds, the supervisor guarantees that (i) no commanded motion toward the goal is produced while d_{min} lies within the stop band; (ii) any resumption is jerk-limited and begins from the same geometric state at which the hold occurred; and (iii) transitions respect hysteresis and dwell, eliminating chatter. These guarantees make the higher-level behaviour legible to an observer and the low-level decisions defensible in analysis.

2.6 Trajectory timing and discrete-time integration

This section specifies how motion references and control are tied to the simulator clock so that approach/hold/resume behaviour is deterministic and auditable. All computations are organised around a single, synchronous timeline shared by MATLAB and CoppeliaSim.

2.6.1 Clocks, rates, and tick semantics

The simulator advances by a fixed physics step T_p. The controller runs with period $T_c = \mathrm{n}T_p$ for some small integer n. One control tick k consists of:

- reading the joint state and tool pose from the simulator at time t_k ,
- evaluating forward kinematics and the geometric Jacobian,
- updating proximity, supervision, and reference sampling,
- composing the joint-velocity command,
- advancing the simulator by n physics steps to t_{k+1}.

All timestamps, logs, thresholds, and timers are expressed on this tick grid; events occur only at tick boundaries. This eliminates hidden latency between what the scene displays and what the controller computes.

2.6.2 Reference generation and sampling

Two reference types are used:

- **Cartesian velocity fields** (non-time-parameterized): a bounded attractive field generates a TCP twist $v_{task}(k)$ that drives the tool toward the goal. Near the goal, a terminal-speed floor prevents asymptotic creep and produces decisive "arrive and stop" behaviour. Velocity caps enforce actuator-compatible magnitudes.
- **Linear-segment-with-parabolic-blend (LSPB) trajectories** (time-parameterized): position and velocity references $x_{ref}(t)$, $\dot{x}_{ref}(t)$ are defined by an acceleration–cruise–deceleration profile with bounds on $||\dot{x}||$and $||\ddot{x}||$ for multi-axis motion, segment times are synchronised so all Cartesian components share a common duration. Sampling is performed strictly on the controller grid: at tick k, the phase s_k (0→1) indexes the LSPB law and produces $x_{ref}(k)$, $\dot{x}_{ref}(k)$. The phase is advanced by a fixed increment per tick unless gated by the supervisor.

To avoid drift, reference phase is accumulated in integer tick units (no fractional time carried across ticks), and quaternion references are re-normalised after interpolation.

2.6.3 Ordering within a tick

Each tick follows a fixed computation order that preserves task priority and numerical correctness:

read → FK/J → proximity update → supervisor gate → reference sample →
compose task twist → form postural bias → project into null space →
saturate & damp → integrate

"Project then integrate" ensures redundancy-confined actions do not leak into the primary task due to discretisation. Damping and velocity caps are applied after composition but before integration, yielding bounded joint increments per tick.

2.6.4 Pause/hold/resume semantics

The supervisor (Section 2.5) gates access to references without altering their geometry:

- **Velocity-field operation**: in hold, the commanded TCP twist is set to zero while the internal integrator state is preserved; on resume, integration restarts from the held pose with a jerk-limited ramp on the speed scale.
- **LSPB operation**: in hold, the reference phase index s_k is frozen; on resume, the remaining segment is executed from the same phase. If the hold straddles a blend boundary, the remainder is re-timed to the tick grid so acceleration and jerk limits are still respected. This produces identical path geometry with inserted dwell and no time-skips.

All ramping (recovery) is quantised to the tick grid and bounded in slope so commanded accelerations remain within limits.

2.6.5 Discrete-time integration and stability guards

Joint-space commands are integrated with a first-order, zero-order-hold scheme over T_c. Three guards keep the integration well behaved:

- **Bounded increments**: joint velocities are capped so that $|\Delta q_i| \leq \dot{q}_{i,max} T_c$. , preventing aliasing of saturation into oscillation.
- **Orientation lock near the goal**: when within an acceptance radius, tool orientation is held to avoid wrist inversions as position errors vanish.
- **Conflict limiter**: when the avoidance bias aligns strongly against the tracking direction and the smallest singular value falls below a threshold, task speed is reduced before integration to avoid large joint excursions in near-singular postures.

2.6.6 Consistent logging

For each tick the logger records: k, t_k. ; the sampled reference $x_{ref}(k)$, $\dot{x}_{ref}(k)$. (or $v_{task}(k)$ in velocity-field runs); the supervisor mode and gate outputs (speed scale, hold flag); the applied joint command; and FK/J diagnostics (singular values, manipulability). By construction, these records are on the same clock as animation and proximity, enabling one-to-one reconstruction of any pause/hold/resume episode.

2.6.7 Determinism and replay

Because all timing derives from T_C and all transitions are tick-synchronised with explicit hysteresis and dwell, repeated runs with the same initial conditions and the same human pose stream produce identical mode sequences and trajectories (modulo floating-point tolerance). This determinism is the basis for the comparative evaluations reported in later chapters.

2.7 Datasets, initial conditions, and scenarios

This section records what varies and what is held fixed across experiments, so that every trajectory, pause/hold/resume episode, and proximity timeline can be reproduced from first principles.

2.7.1 Human-motion traces

Operator motion is provided as time-stamped skeletal pose streams comprising 3D key-points for the major joints. Each trace is (i) trimmed to remove idle pre/post segments, (ii) normalised to the world frame defined in §2.1, and (iii) filtered to suppress jitter while preserving natural limb swing. For experiments, traces are used in two ways:

- **direct replay:** the mannequin in CoppeliaSim is animated frame-by-frame by the normalised skeleton;
- **phase-shifted replay:** the same trace is started at different offsets relative to the robot's approach so that identical motions produce intrusions at distinct points along the task, exercising pause and resume at multiple phases.

The identity of the trace and its phase offset are treated as experimental factors and logged per run.

2.7.2 Initial robot postures

Runs start from a finite set of joint configurations that all realise the same nominal tool pose but differ in elbow/wrist posture. Configurations are generated by solving

the kinematic task with distinct null-space seeds and retaining only those that (i) respect joint limits with margin, (ii) exceed a manipulability threshold, and (iii) satisfy minimum link-to-table clearance. One configuration is designated “neutral” (high manipulability, low joint excursion to the first target); others probe elbow-up/elbow-down and wrist-rotated variants. Selection is either fixed (to compare scenarios like-for-like) or pseudorandom with a recorded seed (to probe sensitivity); in both cases the exact joint vector is logged.

2.7.3 Targets and task geometry

Targets lie inside the tabletop zone introduced in §2.1. Each target is defined by a pose, an acceptance radius, an orientation tolerance, and a nominal approach direction (TCP z at the goal). Keep-out margins at the human edge of the table bound automated approaches. When two targets are used (e.g., move-out/move-back), their poses are chosen so that inter-target motion remains in a dexterous region without posture flips. Target indices and their tolerances are recorded in the run metadata.

2.7.4 Scenario definitions

Five controller configurations exercise the same workcell under progressively richer conditions. For brevity, they are referred to here by their functional roles:

- **Scenario 1:** Target acquisition with a bounded attractive Cartesian velocity field in the absence of an operator;
- **Scenario 2:** Proximity-aware acquisition that halts and holds on intrusion, reshaping posture through redundancy, then resumes when spacing is comfortable;
- **Scenario 3:** Time-parameterised tracking using linear segments with parabolic blends (LSPB) without an operator;
- **Scenario 4:** Supervised pause–resume over an LSPB reference in the presence of proximity events, with deterministic freeze/resume of the trajectory phase;
- **Scenario 5:** Fixed-pose reconfiguration in which the TCP is held while redundancy alone enlarges spacing.

Each scenario inherits the same timing model (§2.6), supervisor (§2.5), and proximity pipeline (§2.3–§2.4). What changes is the reference type (velocity field vs. LSPB), whether proximity events are present, and whether TCP motion is permitted.

2.7.5 Experimental factors and design

Across scenarios, experiments vary along four axes:

- human trace identity and phase (direct vs. phase-shifted replay);
- initial robot posture (neutral vs. alternative null-space realizations);
- target index (single-target approach vs. inter-target motion where applicable);
- proximity thresholds (baseline vs. a slightly tighter set used only for robustness checks).

A small factorial design combines these factors to cover representative operating conditions while keeping the total run count tractable. For sensitivity studies, one factor is swept while others are held fixed at their baseline; ablations toggle individual elements (e.g., hysteresis, orientation lock) to isolate their effect. All random choices are driven by recorded seeds.

2.7.6 Fixed constants

The following items remain invariant within an experimental batch: world→base transform, table geometry and operator-region dimensions, control period and physics step, filter horizons for distance debouncing/rate limits, and the mapping from distance bands to supervisor thresholds/hysteresis/dwell. These constants are declared in a run header and repeated across logs for audit.

2.7.7 Outputs and replay

Every run yields a time-aligned record at the control tick: joint states; TCP pose; sampled references (or task-space twists); global and per-pair distances; supervisor mode and transition events (with reasons and dwell counters); Jacobian singular values, condition number, and manipulability; and joint-saturation flags. Metadata

enumerate the factors above (trace ID/phase, initial posture, target index, scenario role) plus a parameter hash. Replaying a run with the same header, seeds, and assets reproduces the same mode sequence and motion up to floating-point tolerance.

2.8 Assumptions, limitations, and safety envelope summary

This section closes the chapter by making explicit what the workcell model guarantees, what it assumes, and where its scope ends. The aim is to separate the envelope that is enforced by design from the behaviours that are out of scope for this thesis.

2.8.1 Assumptions

- **Environment and agents:** A single human operates on one long side of a bench-top table; a 7-DoF Panda works from the opposite side. The tabletop is planar and unobstructed; tools and fixtures do not change the gross reach geometry during a run.
- **Timing and models:** MATLAB (control/supervision) and CoppeliaSim (geometry/physics) run in synchronous stepping with a fixed control period. The simulator's Panda model is the authoritative source for forward kinematics and Jacobians; joint sensing is idealised (no encoder noise).
- **Human motion signals:** Human pose enters as a time-stamped skeleton stream with bounded jitter. After normalisation and filtering, residual errors and delays are assumed small relative to the control period. Occlusions severe enough to corrupt the skeleton are treated as data faults (see §2.5 fault mode).
- **Proximity representation**: Both agents are approximated for clearance by simple limb-aligned and link-aligned volumes sized conservatively. Monitored link–limb pairs are chosen for realistic interactions at a table; distances to unmonitored pairs are not considered by the supervisor.
- **Control authority:** Joint limits and velocity caps are enforceable at the chosen rates; damping and redundancy-confined posture reshaping can be applied without exciting actuator limits.

- **Task geometry:** Targets lie inside the robot's dexterous zone; acceptance radii and orientation tolerances are specified; a nominal approach direction (TCP z at the goal) is defined for each target.

2.8.2 Safety framework (what is guaranteed by design)

- **Separation governance:** No commanded progression toward the task goal is issued while the global minimum human–robot distance lies inside the stop band. Within the caution band, task motion is conservatively scaled and posture reshaping intensifies; outside, normal tracking proceeds.
- **Deterministic gating:** Transitions between track, caution, hold, recovery obey monotonic escalation, explicit hysteresis, and minimum dwell times tied to the control tick; chattering at thresholds is precluded by construction.
- **Null-space containment:** Clearance-seeking posture changes are confined to redundant directions; primary task motion is unaffected whenever redundancy permits. When redundancy is exhausted (e.g., near limits/singularities), the supervisor, resolves the conflict by slowing or holding.
- **Terminal behaviour:** Near a target, an orientation lock and a terminal-speed floor yield decisive arrivals without wrist inversions. If a hold occurs inside the acceptance radius, resumption continues along the recorded approach direction to the same terminal pose.
- **Auditability:** Every decision is time-aligned to the control clock and logged with the distances, thresholds, dwell counters, and conditioning metrics in effect, enabling reconstruction and review of each slow/hold/resume episode.

2.8.3 Transfer and extension (perspectives)

- **Hardware-in-the-loop:** The synchronous timing and single-source kinematics map directly to real-time middleware; substituting live pose input for recorded streams is the first step toward on-robot trials. See Chapter 8 for the migration roadmap (ROS 2/real-time executors, certified reference governors, and composite SDFs).

- **Governance layers:** The clearance policy can be wrapped by certified reference-governor or safety-programmable logic controllers (PLC) layers to formalise release/hold envelopes against plant-level constraints.
- **Richer models:** The human proxy can be refined (anisotropic limb volumes, tool/workpiece geometry), and multi-sensor fusion can replace single-stream pose input; multi-arm extensions can coordinate null-space policies across robots.

Taken together, these assumptions, limits, and guarantees define the operating envelope for the remainder of the thesis: a reproducible bench-top collaborative cell with clear separation governance, deterministic gating, and auditable behaviour, within which trajectory generation, inverse kinematics, and redundancy management can be evaluated systematically.

Chapter 3

Robotic system kinematic model (Franka Emika Panda)

The model of a robotic arm is a topic that has been extensively addressed in the literature, with well-established formulations for describing geometry, kinematics, and motion generation for industrial manipulators and collaborative robots. Building on this foundation, the present chapter introduces the robotic system adopted in this thesis and the way it is exercised within a shared workspace. The platform is a 7-DoF Franka Emika Panda, a lightweight, redundant manipulator commonly used in human–robot collaboration studies for its accuracy, integrated torque sensing, and ease of integration with simulation and control stacks. Our interest is not limited to the robot as a mechanism, but extends to the collaborative setting in which it operates: a human co-worker, a shared task region, and a control architecture that favors predictable, easily monitored behavior. This aligns with contemporary treatments of redundancy resolution, safety supervision, and human-aware motion as established in the robotics community.

The collaborative cell is realized in CoppeliaSim to mirror an industrial workstation with clear boundaries and observability. The virtual scene comprises a fixed-base Panda mounted on a table, a human work zone represented by a pose-driven avatar, and task objects arranged within the arm's reachable volume. The human motion stream is converted into simplified geometric proxies that allow real-time distance evaluation and separation monitoring without overburdening the control loop. The simulation is synchronized so that sensing, decision, and actuation proceed in lockstep, and the environment is instrumented for continuous logging of the signals that matter for later analysis of throughput, clearance, and task fidelity. In this way, the chapter does not only present a model, but a setting where the model can be exercised repeatedly and transparently.

The intent is to simulate and optimize the robot's behavior under collaborative conditions, increasing the capability to monitor performance and to detect deviations from expected motion or safety margins early. The choices made here

reflect common practice in the field: using a redundant arm to reconcile task objectives and safety, relying on well-known kinematic descriptions and inverse kinematics solvers, and adopting scene abstractions that balance fidelity and computational load. The literature on collaborative robotics, null-space control, and separation monitoring provides the conceptual backdrop for these choices and indicates where they are most applicable in industry: small-batch assembly, inspection, assisted manipulation, and other tasks where a human and a robot share space and responsibilities.

Although developed in simulation, the constructed model and the associated validation method are designed to be replicated on a physical Panda cell with minimal adaptation of frames, limits, and supervisor thresholds. The scene assets, parameters, and procedures are documented to support transfer: frame conventions can be aligned with a real workstation, distance thresholds can be tuned to match sensing hardware, and the same logging and supervision routines can be used to monitor behavior on the floor. This approach ensures that the foundational work reported here can be effectively applied and tested in future implementations, facilitating progression from controlled simulation to pilot deployments and, ultimately, to sustained industrial use.

3.1 Use of the kinematic model in the collaborative cell

This section explains how a kinematic description of a 7-DoF Franka Emika Panda is used to study behavior in a shared workspace with a human. The emphasis is on an operational view rather than derivations. The model is exercised in a CoppeliaSim scene that mirrors an industrial cell: a fixed-base arm on a table, a defined human work zone represented by a pose-driven avatar, and task objects placed within reach. Lightweight link-aligned proxies (capsules/cylinders/spheres) are attached for distance queries and are the only geometry consumed by the supervisor and posture shaper (see Chapter 4). The approach prioritizes predictable motion, clear supervision, and repeatable experiments. The same interfaces and conventions are designed for direct transfer to a physical Panda cell by aligning frames, enforcing the same thresholds, and reusing the synchronized logging routines.

3.1.1 Rationale

A 6×7 kinematic formulation is adopted because collaborative tasks are moderate in speed and benefit from transparency and observability. The robot's internal torque regulation handles low-level dynamics, while the outer loop focuses on where and when the tool moves, and on reorganizing posture to maintain safe threshold distance when a person approaches.

3.1.2 How it is used

The same model supports three recurring situations: tracking simple tool-pose references; tracking with supervised pauses and later resumptions when separation bands are crossed; and a fixed-tool-pose case where only joints move to preserve clearance around the human. The simulation advances in synchronized steps so sensing, decision, and actuation remain aligned, and key signals are logged for later analysis of throughput, transparency, and safety.

3.1.3 Assumptions and scope

The arm is treated as a rigid kinematic chain with calibrated frames and enforced joint limits; self-collision and workspace constraints are respected. Human motion

is converted into simplified geometric proxies in the world frame to enable timely distance evaluation. Full rigid-body dynamics, contact forces, and high-impact interactions are outside scope.

3.1.4 Interfaces referenced later

For consistency across the document, this section introduces the principal quantities used throughout: the tool-pose tracking error, a damped inverse mapping from tool motion to joint motion, the null-space projector that preserves the primary objective while adjusting posture, the minimum robot–human distance computed in the scene, and the stop/release thresholds that govern supervised pauses and resumptions. These definitions establish a common vocabulary for the methods and experiments that follow and will be referenced without further qualification in subsequent chapters; Section 3.4 details the two TCP time-laws (vector vs. LSPB) and their pause/resume semantics; Section 3.5 fixes the safety thresholds used throughout. Section 5.1 details the synchronized logging used to validate these interfaces.

3.2 Robot description

This work employs a 7-DoF Franka Emika Panda (Franka Research 3 generation) mounted as a fixed-base, table-top arm inside a compact collaborative cell in CoppeliaSim. The platform is widely adopted in research labs for human–robot collaboration because it combines human-scale reach, link-side torque sensing on all seven joints, and a research interface that exposes state and control at suitable rates for closed-loop experimentation. At a system level it offers a 3 kg rated payload, ~855 mm reach, and ISO-grade pose repeatability on the order of ±0.1 mm, with joint-space speed limits that support smooth, supervised motion in proximity to a person. These characteristics align with this thesis' emphasis on predictable behavior, clear supervision, and repeatable experiments.

3.2.1 Frames, tool, and workspace

The cell defines a world frame for the scene, a base frame at the Panda mounting, and a tool-center frame at the flange. The arm is installed upright on a bench-height

fixture; task objects are arranged within nominal reach while a dedicated human work zone is kept clear for approach and interaction. This arrangement mirrors typical research and light-industrial layouts and transfers well to hardware because the same frame conventions, mount pose, and safety bands can be reproduced on a physical setup.

3.2.2 Features and platform suitability (research & industry)

- Seven torque-sensorized revolute joints enable compliant behavior and redundancy for posture adjustment in shared workspaces.
- Research interface and ecosystem integrations (ROS 2, MATLAB) facilitate synchronized control, logging, and rapid replication of experiments.
- Certified HRC design and sub-millimetric repeatability support tasks such as small-batch assembly, inspection, assisted manipulation, and teaching by demonstration.

3.2.3 Technical specifications used in this thesis

Table 3.1 consolidates the mechanical and controller-relevant specifications used in this work, including joint ranges, velocity limits, masses, and the manufacturer's repeatable peak torque limits grouped by axes (A1–A2, A3–A4, A5–A7).

Item	Value
Degrees of freedom	7 revolute joints
Rated payload	3 kg
Maximum reach	855 mm
Pose repeatability (ISO 9283)	± 0.1 mm
Typical end-effector speed (limit)	up to ~ 2 m/s
Joint velocity limits (A1-A4 / A5-A7)	$150\%/\text{s}/$ up to $\sim 180 - 301\%$ (per datasheet generation)
Joint position limits (deg)	A1: $= 166 \ldots + 166$; A2: $-101 \ldots + 101$; A3: $-166 \ldots + 166$: A4: $-176 \ldots - 4$; A5: $-166 \ldots + 166$; A6: $-1 \ldots + 215$; A7: $-166 \ldots + 166$
Repeatable peak torque (Nm)	A1: 87; A2: 87; A3: 87; A4: 87; A5: 12; A6: 12; A7: 12.

Table 3.1 Franka Emika Panda arm-level specifications and limits.

Figure 3.1 contextualizes the axis numbering (A1–A7) on the Panda and highlights the repeatable peak-torque limits referenced in Table 3.1.

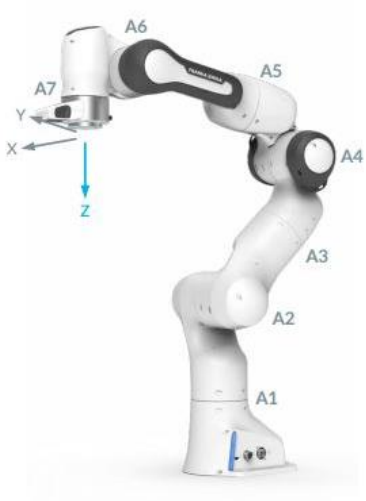

Fig. 3.1 Panda axis map (A1–A7) with manufacturer repeatable peak-torque limits. Axes 1–2: ≤ 87 Nm; axes 3–4: ≤ 87 Nm; axes 5–7: ≤ 12 Nm.

3.2.4 Physical layout and link proxies

For efficient separation monitoring in the simulation, each link is represented by a conservative geometric proxy aligned with its frame; these proxies are used for fast min-distance queries against the human avatar's capsules and to annotate logs with the smallest robot–human clearance. This abstraction keeps computation modest while remaining faithful to the physical envelope and transfers cleanly to hardware deployments (See Chapter 4 for the human model and distance fields).

3.2.5 Kinematic scheme

The Panda is a serial 7R arm: a shoulder with three intersecting joint axes approximating spherical motion, an elbow that extends the reach, and a three-axis wrist that orients the tool. Figure 3.1 sketches the Panda's 7R shoulder–elbow–wrist organization used throughout this chapter.

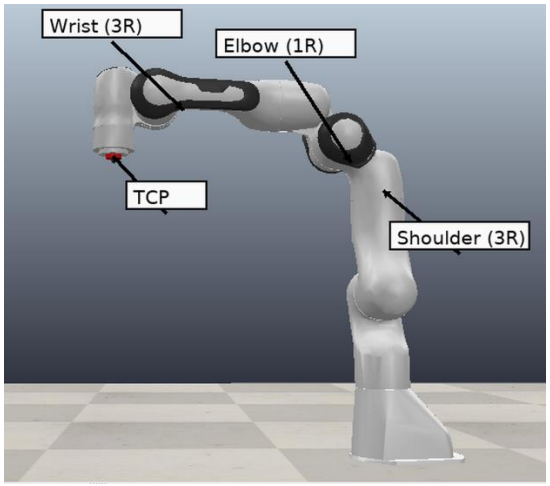

Fig. 3.2 Kinematic scheme of the Panda arm (7R): shoulder with three intersecting axes, elbow extension, and three-axis wrist for tool orientation.

Motions arise from coordinated rotations about each joint axis, producing translation and rotation of the tool in space; redundancy allows posture to be adjusted while the tool pose is maintained within tolerance. This is the working model used throughout the thesis and is consistent with the official robot description files commonly used in research software stacks.

3.2.6 Denavit–Hartenberg (DH) style parametrization adopted in this work

We adopt a modified Denavit–Hartenberg description to fix the link geometry and joint axes in a compact, reproducible way prior to deriving the kinematics. Rather than importing a published table, the parameters used here were fitted to the ground-truth link frames exported from CoppeliaSim at a chosen zero posture and mount. This ensures that forward kinematics reproduce the exact scene used in all experiments. Concretely, the base frame and tool frame were fixed in the simulator; the seven intermediate link frames were exported; and a modified-DH chain $({}^{i-1}A_i(a_i, \alpha_i, d_i, \theta_i)$ with $(\theta_i \equiv q_i)$ was solved so that cumulative transform $(\mathrm{T}_{\mathrm{i}}^{(0)} = \prod_{\mathrm{k}=1}^{\mathrm{i}} A_k^{(k-1)})$ aligned (within numerical tolerance) with the exported frames at the zero posture. A fixed flange transform $({}^{7}T_T)$ was then set to match the desired tool offset. This fit was validated by checking that (i) forward kinematics at random configurations matched the simulator exports to within a small positional and angular error, and (ii) the base–tool transform remained consistent when the same chain was driven by scene joint values.

For $i = 1, 2, \ldots, 7$ we use a modified-DH link transform with joint variable $\theta_i \equiv q_i$ and constants (a_i, α_i, d_i):

$$ {}^{i-1}A_i(q_i) = \begin{bmatrix} R_z(\theta_i)R_x(\alpha_i) & R_z(\theta_i)\begin{bmatrix} a_i \\ 0 \\ d_i \end{bmatrix} \\ 0 & 1 \end{bmatrix} $$

The cumulative transforms and TCP pose are

$$ {}^{0}T_i(q) = \prod_{k=1}^{i} {}^{k-1}A_k(q_k), \qquad {}^{0}T_{\mathcal{T}}(q) = {}^{0}T_7(q)\,{}^{7}T_{\mathcal{T}} $$

with fixed tool offset: ${}^{7}T_{\mathcal{T}}: d_f = 0.107\ m$.

The constants used are:

i	a_i[m]	α_i[rad]	d_i[m]
1	0	0	0.333
2	0	$-\frac{\pi}{2}$	0
3	0	$+\frac{\pi}{2}$	0.316
4	0.0825	$+\frac{\pi}{2}$	0
5	-0.0825	$-\frac{\pi}{2}$	0.384
6	0	$+\frac{\pi}{2}$	0
7	0.088	$+\frac{\pi}{2}$	0

Table 3.2 DH constant parameteres.

3.2.7 Jacobian (formulation & components, as used)

The geometric Jacobian is the linear operator that maps joint-rate space to the instantaneous twist of the tool frame. Figure 3.2 illustrates the construction of Jacobian columns from joint axes and point positions.

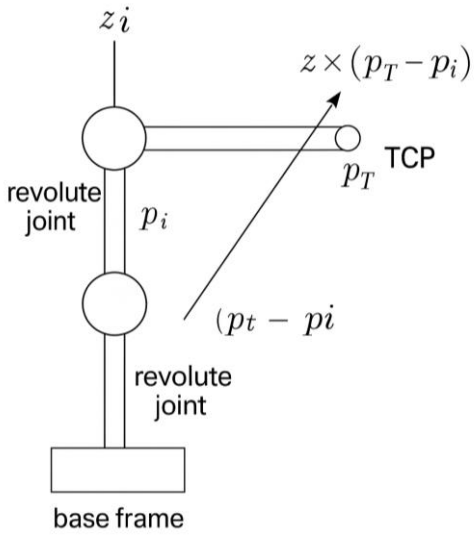

Fig. 3.3 For a revolute joint i, $J_{v,i} = \mathbf{z}_i \times (\mathbf{p}_T - \mathbf{p}_i)$ and $J_{\omega,i} = \mathbf{z}_i$.Vectors are resolved in frame 0 as used in the implementation.

Let $p_i \in R^3$ and $z_i \in R^3$ denote, respectively, the origin and unit z-axis of frame i expressed in $\beta = 0$, extracted from ${}^0T_i(q)$. Let p_T be the origin of the TCP from ${}^0T_T(q)$ For a revolute joint i the $\mathrm{i}-\mathrm{th}$ column of the geometric Jacobian resolved in 0 is:

$$J_{v,i} = z_i \times (p_T - p_i)\,, \qquad J_{\omega,i} = z_i$$

Stacking columns yields

$$J^0(q) = \begin{bmatrix} J_{v,1} & \cdots & J_{v,7} \\ J_{\omega,1} & \cdots & J_{\omega,7} \end{bmatrix} \in \mathbb{R}^{6\times 7}$$

When the Jacobian is required in the TCP frame $\mathcal{T}$ we apply the rigid rotation with $R_{\mathcal{T}} = {}^0R_{\mathcal{T}}(q)$:

$$J^{\mathcal{T}}(q) = \begin{bmatrix} R_{\mathcal{T}}^{\top} & 0 \\ 0 & R_{\mathcal{T}}^{\top} \end{bmatrix} J^0(q)$$

These expressions are exactly those implemented: transforms are formed from the DH-style chain consistent with the scene $(p_i,\, z_i,\, p_T)$ are extracted, each column is assembled via the cross-product rule above, and an optional frame change, yields $J^{\{T\}}$. A different tool is incorporated by updating ${}^7T_T$ before computing p_T.

3.3 Mathematical model of the 7-DoF arm

This section establishes the mathematical description used to command and assess the Franka Emika Panda in the collaborative cell. The objective is to fix a consistent kinematic model, specify how tool pose and errors are represented, and define the velocity mapping between joint space and task space that underpins all experiments. Starting from the scene-consistent link frames introduced earlier, we derive forward kinematics and a pose-error definition suited to small, well-conditioned corrections. We then formalize the geometric Jacobian and the frame conventions used when relating joint rates to tool twist. Building on these, we present the inverse kinematics solver based on damped least squares with SVD regularization, together with the adaptive damping and saturation policies that keep commands within safe bounds. Measures of conditioning and manipulability are introduced to identify neighborhoods where stronger regularization is required. Finally, we state the task-priority composition used to preserve the primary tool objective while shaping joint-space behavior in the null space, including a leak guard to monitor task preservation, and outline the orientation-locking strategy employed when the tool pose must remain fixed. The intent is to provide a clear, self-contained reference for the methods implemented in the collaborative scenarios that follow.

3.3.1 Forward kinematics and pose-error definition

Before specifying control laws, we fix how the arm's pose is computed and how deviations from a desired pose are measured. The forward kinematic map provides a unique tool pose for each joint configuration, and the error representation must remain well behaved under the small, incremental motions characteristic of supervised collaboration. The conventions below follow the scene-consistent link frames defined earlier so that all computations match the simulated cell one-to-one.

Let ${}^{0}T_{i}(q) \in SE(3)$ be the cumulative transform from the base/world frame $\{0\}$ to link i, constructed from the modified-DH chain fitted to the CoppeliaSim frames in Section 3.2. The TCP (tool) pose is

$$ {}^{0}T_{\mathcal{T}}(q) = \begin{bmatrix} {}^{0}R_{\mathcal{T}}(q) & {}^{0}p_{\mathcal{T}}(q) \\ 0 & 1 \end{bmatrix} = {}^{0}T_{7}(q)\,{}^{7}T_{\mathcal{T}} $$

where ${}^{7}T_{\mathcal{T}}$ is the fixed flange-to-tool transform.
Given a desired TCP pose

$$ {}^{0}T_{\mathcal{T}}^{\text{des}} = \begin{bmatrix} {}^{0}R_{\mathcal{T}}^{\text{des}} & {}^{0}p_{\mathcal{T}}^{\text{des}} \\ 0 & 1 \end{bmatrix} $$

the translational error (resolved in $\{0\}$) is

$$ e_p = {}^{0}p_{\mathcal{T}}^{\text{des}} - {}^{0}p_{\mathcal{T}}(q) \in \mathbb{R}^3 $$

For the rotational component, we adopt an axis-angle representation derived from the right-invariant rotation error

$$ R_{\text{err}} = {}^{0}R_{\mathcal{T}}^{\text{des}}({}^{0}R_{\mathcal{T}}(q))^{\top} \in SO(3) $$

The orientation error vector is the matrix logarithm of R_{err} ,

$$ e_\omega = \log(R_{\text{err}}) \in \mathbb{R}^3, $$

i.e., the unique rotation vector whose direction is the principal axis of R_{err} and whose magnitude $\theta \in [0, \pi]$ is the principal angle. When $R_{\text{err}} \neq I$, a closed-form evaluation is

$$ \theta = \cos^{-1}\left(\frac{\text{tr}(R_{\text{err}}) - 1}{2}\right), \qquad u = \frac{1}{2\sin\theta}\begin{bmatrix} R_{\text{err}}(3,2) - R_{\text{err}}(2,3) \\ R_{\text{err}}(1,3) - R_{\text{err}}(3,1) \\ R_{\text{err}}(2,1) - R_{\text{err}}(1,2) \end{bmatrix}, $$

$$ e_\omega = \theta u $$

and $e_\omega = 0$ when $R_{\text{err}} = I$. The branch of θ is selected to preserve continuity for small attitude corrections; in implementation $\|e_\omega\|$ is limited to remain within the injectivity radius.

The pose-error vector used by the velocity-level controller stacks translation and rotation as

$$ e_x = \begin{bmatrix} e_p \\ e_\omega \end{bmatrix} \in \mathbb{R}^6, $$

resolved in $\{0\}$ unless stated otherwise. This representation avoids Euler-angle singularities, remains well conditioned for small corrections, and aligns with the frame and tolerance conventions fixed in Chapter 3.1.

3.3.2 Geometric Jacobian and frame conventions implementation

Before introducing inversion or task composition, we pin down exactly how the Jacobian is constructed in this work so that the matrix used by the solver matches the geometry of the CoppeliaSim scene one-to-one. The goal is a reproducible pipeline: take the scene's link frames, apply the fitted modified-DH chain, and assemble a 6×7 Jacobian whose columns have a clear physical meaning and a declared frame resolution.

Input and resolution

The function takes the joint vector $q \in \mathbb{R}^7$ and a fixed flange-to-tool transform ${}^{7}T_{\mathcal{T}}$. Unless otherwise stated, all intermediate quantities are expressed in the base/world frame $\{0\}$. When needed, the Jacobian is rotated to the tool frame $\{\mathcal{T}\}$.

Step 1 - Forward kinematics consistent with the scene

Using the modified-DH chain fitted to the exported link frames (§3.2), form the cumulative transforms

$$ {}^{0}T_{i}(q) = \prod_{k=1}^{i} {}^{k-1}A_{k}(q_k), i = 1, \dots, 7 $$

and the TCP pose

$$ {}^{0}T_{\mathcal{T}}(q) = {}^{0}T_{7}(q)\, {}^{7}T_{\mathcal{T}} = \begin{bmatrix} {}^{0}R_{\mathcal{T}}(q) & {}^{0}p_{\mathcal{T}}(q) \\ 0 & 1 \end{bmatrix} $$

The DH chain is used solely to generate consistent transforms; the geometric Jacobian is assembled from frame axes and positions (cross-product rule) to avoid DH-specific pitfalls.

Step 2 - Extract per-joint geometric primitives

From each ${}^{0}T_{i}(q)$ extract:

$p_i \in \mathbb{R}^3$ (origin of frame i in $\{0\}$), $z_i \in \mathbb{R}^3$ (unit z-axis of frame i in $\{0\}$).

Also take $p_{\mathcal{T}} = {}^0p_{\mathcal{T}}(q)$.

Step 3 - Assemble columns (all joints revolute)

For each joint $i \in \{1, \dots, 7\}$, the column of the geometric Jacobian resolved in $\{0\}$ is

$$J_{v,i} = z_i \times (p_{\mathcal{T}} - p_i), \qquad J_{\omega,i} = z_i$$

Stacking gives

$$J^{\{0\}}(q) = \begin{bmatrix} J_{v,1} & \cdots & J_{v,7} \\ J_{\omega,1} & \cdots & J_{\omega,7} \end{bmatrix} \in \mathbb{R}^{6\times 7}, \qquad \dot{x}^{\{0\}} = J^{\{0\}}(q)\dot{q}$$

When a TCP-resolved Jacobian is required, we apply a rigid rotation to map columns from $\{0\}$ to $\{\mathcal{T}\}$, keeping the assembly numerically identical but frame-consistent.

Step 4 - Optional change of resolution to the tool frame

When a tool-resolved twist is required, apply the current TCP rotation

$R_{\mathcal{T}} = {}^0R_{\mathcal{T}}(q)$

$$J^{\{\mathcal{T}\}}(q) = \begin{bmatrix} R_{\mathcal{T}}^{\top} & 0 \\ 0 & R_{\mathcal{T}}^{\top} \end{bmatrix} J^{\{0\}}(q), \qquad \dot{x}^{\{\mathcal{T}\}} = J^{\{\mathcal{T}\}}(q)\dot{q}$$

Step 5 - Scene-alignment and tool changes

The flange/tool transform ${}^7T_{\mathcal{T}}$ can be swapped to represent a different end-effector without changing any formula: it only shifts $p_{\mathcal{T}}$ and rotates the resolution if $J^{\{\mathcal{T}\}}$ is requested. Because the DH constants were fitted to the scene's zero posture, the (p_i, z_i) extracted here align with the simulator frames at all configurations.

Step 6 - Consistency checks used in this work

At random configurations, verify finite-difference consistency:

$$\frac{{}^{0}p_{\mathcal{T}}(q+\varepsilon e_j)-{}^{0}p_{\mathcal{T}}(q)}{\varepsilon}\approx J_{v,j},\qquad \frac{\log\left({}^{0}R_{\mathcal{T}}(q)^{\top}\,{}^{0}R_{\mathcal{T}}(q+\varepsilon e_j)\right)}{\varepsilon}\approx J_{\omega,j},$$

for small $\varepsilon > 0$ and standard basis e_j. Also check the frame-change identity

$$\begin{bmatrix} R_{\mathcal{T}} & 0 \\ 0 & R_{\mathcal{T}} \end{bmatrix} J^{\{\mathcal{T}\}}(q) \approx J^{\{0\}}(q)$$

up to numerical tolerance. The chosen resolution (base or tool) is recorded with each run to avoid ambiguity in later analyses.

Physical reading of the columns (as implemented)

Each column encodes the screw motion induced at the TCP by an infinitesimal rotation of joint i : the lower block z_i is the angular part (about the joint axis), and upper block $z_i \times (p_{\mathcal{T}} - p_i)$ is the linear part (lever-arm effect of that axis at the TCP). Proximal joints contribute strongly to both translation and orientation; distal joints primarily trim orientation and fine positioning—exactly what is observed in the experiment logs.

3.3.3 Damped least-squares inverse with SVD, adaptive damping, and saturations

Before composing tasks, we fix the velocity-level inverse kinematics used throughout. Figure 3.4 summarizes the DLS–SVD inverse used throughout, including adaptive damping and safety saturations.

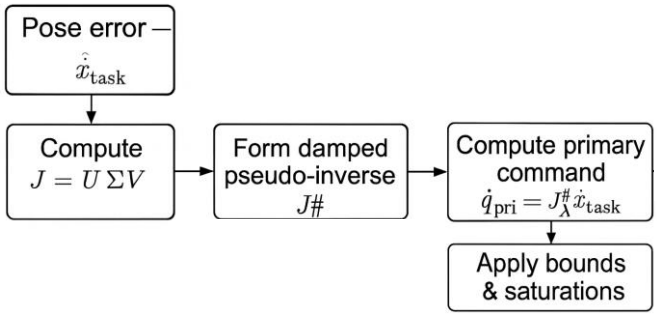

Fig. 3.4 DLS–SVD pipeline: compute $J = U\Sigma V^{\top}\rightarrow$ form $J_{\lambda}^{\#}\rightarrow$ primary command $\dot{q}_{\text{pri}} = J_{\lambda}^{\#}\dot{x}_{\text{task}}\rightarrow$ apply bounds and saturations.

The objective is a numerically stable joint-rate command that behaves predictably near poor conditioning while respecting joint limits. Given the geometric Jacobian $J(q) \in \mathbb{R}^{6\times7}$, with $q \in \mathbb{R}^7$ the joint configuration, we compute its singular value decomposition $J = U\Sigma V^\top$, where $U \in \mathbb{R}^{6\times6}$ and $V \in \mathbb{R}^{7\times7}$ are orthogonal, and $\Sigma = \mathrm{diag}(\sigma_1, \dots, \sigma_6)$ collects the nonnegative singular values σ_i. The damped least-squares (DLS) pseudoinverse is then defined as

$$J_\lambda^{\#} = V\mathrm{diag}\left(\frac{\sigma_i}{\sigma_i^2 + \lambda^2}\right)U^\top$$

with $\lambda > 0$ a damping parameter that regularizes the inversion in directions associated with small σ_i. For a desired task-space velocity $\dot{x}_{\text{task}} = [v^\top \omega^\top]^\top \in \mathbb{R}^6$ (linear part v in $\mathrm{m/s}$ and angular part ω in rad $/\mathrm{s}$, resolved in the chosen frame), the primary joint command is

$$\dot{q}_{\text{pri}} = J_\lambda^{\#}\dot{x}_{\text{task}},$$

which is the minimum-norm joint-rate vector that best realizes $\dot{x}_{\text{task}}$ under the damping λ.

To make the inverse robust across the workspace, we adapt λ to the instantaneous conditioning. We monitor the condition number $\kappa(J) = \sigma_{\max}/\sigma_{\min}{}^{+}$, where $\sigma_{\max}$ is the largest singular value and $\sigma_{\min}{}^{+}$is the smallest nonzero singular value encountered at q. The damping is scheduled as

$$\lambda(q) = \lambda_{\min} + (\lambda_{\max} - \lambda_{\min})s\left(\frac{\kappa(J) - \kappa_{\text{ok}}}{\kappa_{\text{hi}} - \kappa_{\text{ok}}}\right)$$

where $\lambda_{\min}, \lambda_{\max}$ bound the admissible damping, $\kappa_{\text{ok}} < \kappa_{\text{hi}}$ mark the transition from well-conditioned to poorly conditioned regions, and $s(\cdot) \in [0,1]$ is a smooth clamping function (e.g., cubic or logistic) that blends between the bounds. Thus λ remains close to $\lambda_{\min}$ in favorable regions and increases toward $\lambda_{\max}$ as $\kappa(J)$ grows.

The task-space demand $\dot{x}_{\text{task}}$ itself is shaped from the pose error of 3.3.1 using bounded proportional action,

$$\dot{x}_{\text{task}} = \begin{bmatrix} K_p e_p \\ K_\omega e_\omega \end{bmatrix}$$

where $e_p \in \mathbb{R}^3$ (meters) and $e_\omega \in \mathbb{R}^3$ (axis-angle, radians) are the position and orientation errors resolved in the same frame as the Jacobian, K_p and K_ω are diagonal nonnegative gain matrices, and componentwise clamps enforce $\|K_p e_p\| \leq v_{\max}$ and $\|K_\omega e_\omega\| \leq \omega_{\max}$ so that linear and angular rate caps $v_{\max}$ (m/s) and $\omega_{\max}$ (rad/s) are never exceeded before inversion [1, 2].

For later null-space composition we define the projector

$$N(q) = I_7 - J_\lambda^{\#}(q) J(q)$$

where I_7 is the 7×7 identity. This operator removes any component of a joint-rate vector that would leak into the primary task, allowing secondary behaviors to be added without corrupting $\dot{x}_{\text{task}}$ (see §3.5.5).

Leakage is monitored as $\ell = \|J(q)\, N(q)\, \dot{q}\ \text{bias}\|$ and clamped below LEAK_THR by scaling the secondary command. The leak metric, scale factor, and saturation flags are logged every tick for auditability (see §3.5.7).

Finally, after adding secondary terms and obtaining a provisional $\dot{q}$, we enforce joint-rate limits uniformly. Let $|\dot{q}|_{\max,i}$ be the admissible speed (rad/s) for joint i. If $|\dot{q}_i| > |\dot{q}|_{\max,i}$, we rescale

$$\dot{q} \leftarrow \gamma \dot{q}, \quad \gamma = \min\left(1, \min_i \frac{|\dot{q}|_{\max,i}}{|\dot{q}_i| + \varepsilon}\right),$$

with a small $\varepsilon > 0$ to avoid division spikes when $|\dot{q}_i| \approx 0$. This preserves the command direction while guaranteeing all joints satisfy their caps. As an additional numerical safeguard, singular values below a small floor $\sigma_{\min}$ are replaced by $\tilde{\sigma}_i = \max(\sigma_i, \sigma_{\min})$ before forming the diagonal factors $\sigma_i/(\sigma_i^2 + \lambda^2)$, which reduces jitter in directions that are effectively uncontrollable. In practice we recompute the SVD only when changes in $\kappa(J)$ or $\|e_x\|$ exceed small hysteresis thresholds, and we log the triplet $(\lambda, \kappa(J), \gamma)$ each control cycle together with the Jacobian's resolution (base or tool) for traceability.

3.3.4 Manipulability, conditioning and safe neighborhoods

Before composing primary and secondary behaviors, we delineate where the kinematic map is reliable and how the controller responds as conditioning degrades. The objective is twofold: quantify local dexterity in a way that is reproducible from logs, and embed guardrails that keep inversion well-behaved without unnecessarily slowing motion. All metrics here are computed from the same geometric Jacobian $J_{(q)}$ and its singular value decomposition $J = U\Sigma V^{\top}$ introduced earlier, with singular values $\sigma_1 \geq \cdots \geq \sigma_6 \geq 0$ defining the principal task-space directions.

We use three complementary indicators. First, the condition number $\kappa(J) = \sigma_{\max}/\sigma_{\min}$ (where $\sigma_{\min}$ (is the smallest nonzero singular value at q) captures worst-case anisotropy of the velocity map; it grows unbounded near singular configurations and is therefore effective for triggering stronger regularization. Second, the Yoshikawa manipulability index $w(q) = \sqrt{\det(JJ^{\top})} = \prod_{i=1}^{6} \sigma_i$ measures the hyper-volume of attainable twists per unit joint-rate norm; it collapses to zero at singularities and is sensitive to simultaneous shrinkage of several directions rather than just the smallest one. Third, a directional measure useful for experiments is the reciprocal gain along a desired twist direction $u \in \mathbb{R}^6$ (with $\|u\| = 1$): we define $m_u(q) = 1/\left\|J_{\lambda}^{\#}(q)u\right\|$. This quantity reports how much joint motion would be required to realize a unit-magnitude command along u; small m_u flags directions that are expensive or poorly controllable even when the aggregate indices still look acceptable. In practice we log $\kappa(J), w(q)$, and a small set of m_u aligned with the commanded twist to make the diagnosis of slowdowns unambiguous.

These indicators ground the definition of safe neighborhoods. We specify two nested sets with hysteresis: a nominal region $\mathcal{S}_{\text{ok}} = \{q: \kappa(J) \leq \kappa_{\text{ok}} \wedge w(q) \geq w_{\text{ok}}\}$ in which the inverse operates at low damping and full task gains, and a guarded region $\mathcal{S}_{\text{guard}} = \{q: \kappa(J) \leq \kappa_{\text{hi}} \wedge w(q) \geq w_{\text{lo}}\}$ that extends $\mathcal{S}_{\text{ok}}$ by a margin. Entering $\mathcal{S}_{\text{guard}} \setminus \mathcal{S}_{\text{ok}}$ increases the damping $\lambda(q)$ according to the schedule in §3.3.3 and proportionally reduces the translational and angular gains used to build $\dot{x}_{\text{task}}$ (so both the inversion and the prefilter cooperate). If either bound is violated ($\kappa(J) > \kappa_{\text{hi}}$ or $w(q) < w_{\text{lo}}$), task gains are clipped to minimal values and the null-

space term is restrained to prevent the controller from "pushing into" a singularity while trying to improve posture. The hysteresis ($\kappa_{\mathrm{ok}} < \kappa_{\mathrm{hi}}, w_{\mathrm{lo}} < w_{\mathrm{ok}}$) prevents chatter as the arm hovers near the boundary; exact thresholds are reported alongside results so experiments are reproducible.

Two implementation details improve fidelity to the physical robot and comparability across tasks. First, when translation and rotation have very different operational scales, we optionally introduce a task metric $W = \mathrm{diag}(s_p I_3, s_\omega I_3)$ that re-weights the twist before inversion by replacing J with $W^{1/2}J$ in the SVD; the scalars $s_p > 0$ and $s_\omega > 0$ set translation-rotation balance without altering frame conventions. All logged indices are then computed on the weighted Jacobian so that the reported conditioning matches the controller's internal view. Second, proximity to joint limits can degrade effective dexterity even when $\kappa(J)$ is moderate; to capture this we monitor a joint-margin factor:

$\rho(q) = \min_i \{(q_{\max,i} - q_i)/(q_{\max,i} - q_{\min,i}), (q_i - q_{\min,i})/(q_{\max,i} - q_{\min,i})\} \in [0,0.5]$.

When $\rho(q)$ drops below a comfort bound, the bias term in §3.3 .5 is directed away from the nearest limit and the admissible null-space velocity is reduced, which in turn helps keep $w(q)$ from collapsing in subsequent steps.

Finally, all quantities in this subsection are resolved consistently with the Jacobian's chosen frame (base or tool) and are evaluated at the same discrete-time index as the SVD used for inversion. We record $\kappa(J)$, $w(q), \{m_u\}, \rho(q)$, the active thresholds, and the resulting gains and $\lambda(q)$ per control cycle. This establishes an auditable link from the reported performance—e.g., pauses or slowdowns near corners of the workspace—to the numerical state of the kinematic map at the time decisions were made.

3.3.5 Task–priority composition and leak guard

Having fixed the forward map and a robust inverse, we now describe how primary tool-space objectives are preserved while secondary joint-space behaviors reshape posture, respect limits, and create clearance around the human. The construction follows the classical task–priority paradigm but is specialized to the scene-

consistent Jacobian and damped inverse introduced earlier, with explicit guards to prevent priority violations and to make behavior auditable from logs.

We denote by $\dot{x}_{\text{task}} \in \mathbb{R}^6$ the bounded twist demand assembled from the pose error, using the gains and rate caps defined in §3.3.3. The primary joint command associated with this demand is obtained via the damped pseudoinverse $J_\lambda^{\#}(q)$ of the geometric Jacobian $J(q)$:

$$\dot{q}_{\text{pri}} = J_\lambda^{\#}(q)\dot{x}_{\text{task}}\,.$$

To embed secondary behaviors without corrupting the primary objective, we project them through the null space of the current Jacobian. With

$$N(q) = I_7 - J_\lambda^{\#}(q)J(q),$$

any joint-rate vector of the form $N(q)\eta$ leaves the instantaneous primary twist unchanged, because $JN = 0$ by construction (up to damping-induced numerical residue). The complete command therefore reads

$$\dot{q} = \underbrace{J_\lambda^{\#}\dot{x}_{\text{task}}}_{\text{primary}} + \underbrace{N\dot{q}_{\text{bias}}}_{\text{secondary}},$$

where $\dot{q}_{\text{bias}} \in \mathbb{R}^7$ aggregates joint-space terms that encode posture preferences, joint-limit avoidance, and safety-driven reconfiguration. In this thesis we use smooth, bounded ingredients so that $\dot{q}_{\text{bias}}$ remains interpretable and differentiable: (i) a posture term pulling toward a comfortable reference q_*, (ii) a joint-limit barrier that increases as any joint approaches $q_{\min}$ or $q_{\max}$, and (iii) a safety field steering links away from the human when distances decrease. These are combined with positive weights that may depend on the current context (e.g., larger weight on limit avoidance when margin shrinks), but always pass through the same projector N to guarantee priority.

Two practical issues must be addressed to make this composition predictable in the collaborative cell. First, because $J_\lambda^{\#}$ is damped, the identity $JN = 0$ holds only up to a small numerical residue; if $\dot{q}_{\text{bias}}$ is large, that residue can leak into the primary channel. We therefore monitor the instantaneous leakage

$$\ell(q, \dot{q}_{\text{bias}}) = \|J(q)N(q)\dot{q}_{\text{bias}}\|,$$

resolved in the same frame as J, and enforce $\ell \leq$ LEAK_THR by scaling $N\dot{q}_{\text{bias}}$ when necessary. This single scalar, logged at each control cycle, makes priority violations observable and gives an immediate diagnostic for cases where a strong secondary push would otherwise disturb the tool objective. Second, because both the Jacobian and the projector vary with configuration, fast changes in $\dot{q}_{\text{bias}}$ can create chattering if they are allowed to react instantaneously to small distance or margin fluctuations. To avoid this, we employ hysteresis and dwell: the weights inside $\dot{q}_{\text{bias}}$ change only after the corresponding signal crosses a threshold with a small margin (e.g., distance bands for human proximity, comfort bands for joint margins), and then remain fixed for a minimum dwell time before they can move back. This simple policy significantly improves smoothness without sacrificing responsiveness.

Finally, the composite command $\dot{q}$ inherits the safety policies from §3.3.3. If any per-joint speed bound would be exceeded, a uniform scale is applied to the entire vector so that all components respect their caps while preserving direction. The resolution (base or tool) of the primary twist and the Jacobian is recorded with the same timestamp as ℓ, the active weights in $\dot{q}_{\text{bias}}$, and the global scale factor, ensuring that every experiment can be traced back from observed TCP behavior to the precise state of the priority stack at that moment.

3.3.6 Orientation locking for the fixed-TCP scenario

In collaborative operation there are phases where the tool must remain immobile in space while the arm reconfigures around the human. This subsection specifies the fixed-TCP regime as implemented: how the controller holds a constant tool pose, how residual motion is bounded and monitored, and how null-space reconfiguration proceeds without degrading the primary objective.

Lock objective and admissible drift

At the onset of the regime, the controller captures the tool pose

$$^{0}T_{\mathcal{T}}^{\text{lock}} = \left[\, ^{0}R_{\mathcal{T}}^{\text{lock}} \mid ^{0}p_{\mathcal{T}}^{\text{lock}} \,\right]$$

and regulates the instantaneous error

$$e_p(t) = {}^0p_{\mathcal{T}}^{\text{lock}} - {}^0p_{\mathcal{T}}(q(t)), \qquad e_\omega(t) = \log\left({}^0R_{\mathcal{T}}^{\text{lock}}\ {}^0R_{\mathcal{T}}(q(t))^\top\right)$$

The controller enforces the tight bounds

$$\|e_p\| \leq \text{POS\_TOL}, \qquad \|e_\omega\| \leq \text{ROT\_TOL}$$

and flags a lock-violation whenever either bound is exceeded; the correction is applied immediately and the event is logged.

Figure 3.5 depicts the captured lock pose and the bounded error region enforced during fixed-TCP reconfiguration:

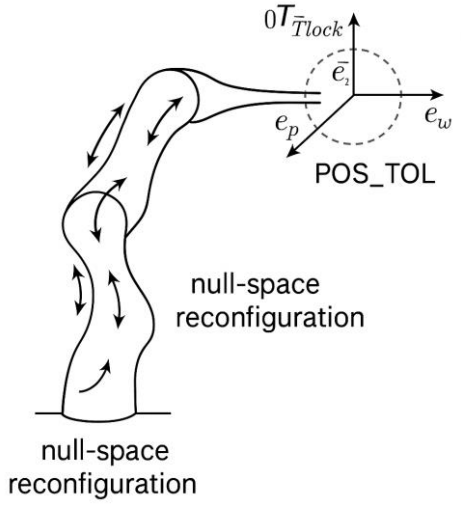

Fig. 3.5 Fixed-TCP regime. The controller captures ${}^0T_{\mathcal{T}}^{\text{lock}}$ and enforces $\| e_p \| \leq$ POS_TOL , $\| e_\omega \| \leq$ ROT_TOL while null-space reconfiguration proceeds.

Primary command (near-zero set-point)

The task twist is a clipped proportional action that recenters the TCP on the lock pose while keeping commanded motion negligible:

$$\dot{x}_{\text{lock}} = \begin{bmatrix} K_p^{\text{lock}}\,\text{sat}\left(e_p; v_{\max}^{\text{lock}}\right) \\ K_\omega^{\text{lock}}\,\text{sat}\left(e_\omega; \omega_{\max}^{\text{lock}}\right) \end{bmatrix}$$

with small diagonal gains $K_p^{\text{lock}}, K_\omega^{\text{lock}} \geq 0$ and stringent caps $v_{\max}^{\text{lock}} \ll v_{\max}$, $\omega_{\max}^{\text{lock}} \ll \omega_{\max}$. The corresponding joint command is

$$\dot{q}_{\text{pri}} = J_{\lambda}^{\#}(q)\dot{x}_{\text{lock}},$$

using the same DLS inverse and adaptive damping as in §3.3.3. This continuously suppresses drift accumulated from numerical residue or sensor noise.

Null-space reconfiguration under a lock

Secondary behavior is admitted exclusively through the projector $N(q) = I_7 - J_{\lambda}^{\#}(q)J(q)$:

$$\dot{q} = \underbrace{J_{\lambda}^{\#}\dot{x}_{\text{lock}}}_{\text{pose hold}} + \underbrace{N\dot{q}_{\text{bias}}}_{\text{reconfiguration}}$$

where $\dot{q}_{\text{bias}}$ aggregates posture regulation, joint-limit margins, and safety-field repulsion. Because damping introduces a small numerical residue, the controller monitors the instantaneous leakage

$$\ell = \|J(q)N(q)\dot{q}_{\text{bias}}\|$$

in the same resolution as J and scales $N\dot{q}_{\text{bias}}$ to enforce $\ell \leq$ LEAK_THR. Leakage, scale factor, and bounds compliance are recorded every cycle.

Lock variants (implemented)

Two variants are implemented and used in experiments:

- **Full-pose lock (default):** the 6-D $\dot{x}_{\text{lock}}$ above holds both position and attitude within POS_TOL and ROT_TOL.
- **Orientation-only lock (ablations):** the solver holds attitude rigidly while allowing millimetric position accommodation. This is realized with a selection matrix $S = \text{diag}(0 \cdot I_3, I_3)$, applied as $\dot{x}_{\text{lock}} \leftarrow S\left[K_p^{\text{lock}}\, e_p; K_{\omega}^{\text{lock}}\, e_{\omega}\right]$ and $J \leftarrow SJ$.

Orientation-only lock is used in ablations; the full-pose lock is the default for fixed-TCP runs reported in Chapter 6.

Anti-drift measures and logging

Axis-angle errors are clamped to the injectivity radius; $\dot{x}_{\text{lock}}$ is low-pass filtered with a short time constant; integral action is disabled in null-space terms within a guard band around the lock. The controller reports the drift metrics

$$\delta_p(t) = \left\| e_p(t) \right\|, \qquad \delta_\omega(t) = \| e_\omega(t) \|$$

together with $\ell(t)$ and the applied scale on $N\dot{q}_{\text{bias}}$. All quantities are resolved consistently with the Jacobian (base or tool frame) and time-aligned with the IK solve, providing an auditable record that ties TCP immobility to the instantaneous numerical state of the lock controller.

3.4 Trajectory time law for the TCP (Vector vs. LSPB)

This section defines how the tool-center-point (TCP) reference is generated in time under two alternative laws that are used throughout the experiments: a continuous vector-attractive field and a lane-standard linear–segment–parabolic–blend (LSPB) profile. Both laws produce feasible, smooth target twists for the kinematic controller, but they emphasize different priorities. The vector field favors immediacy and reactivity to changing goals and safety cues, offering a memoryless reference that can be redirected at any instant with minimal timing structure. LSPB, in contrast, prescribes an explicit acceleration–cruise–deceleration envelope with axis synchronization, bounded jerk at blend transitions, and well-defined start/stop timing; it is therefore the natural choice when the supervisor must pause and later resume progress without corrupting the intended schedule.

The presentation is deliberately operational and consistent with the rest of the chapter: references are expressed in the world frame at the TCP, sampled on the controller tick, and shaped so that commanded linear and angular rates remain within the limits enforced by the kinematic layer. We first formalize the vector-attractive baseline and discuss its responsiveness and lack of temporal guarantees, then derive the discrete-time LSPB with synchronization and pause/resume semantics compatible with the speed-and-separation monitor. We conclude with the constraint-enforcement mechanisms common to both laws (velocity/acceleration

caps and command saturation) and specify the figures and tables used later for reproducibility.

This section concerns TCP time laws used in the moving-TCP modes. In the fixed-TCP mode, the TCP reference is held constant by §3.3.6, and only a null-space joint motion is generated; hence no TCP time-parameterization is required here.

3.4.1 Vector-field TCP reference (moving target → repel → fixed target)

This subsection keeps the continuous, signal-driven reference but states the exact law used to produce the TCP twist each control tick. The aim is to (i) approach a moving target while it is in motion, (ii) produce immediate retreat when separation must increase, and (iii) resume convergence to a fixed goal once RELEASE is granted, all without re-planning or retiming.

Let the instantaneous desired TCP pose be

$$ {}^{0}T_{T}^{\mathrm{des}}(t) = \begin{bmatrix} {}^{0}R_{T}^{\mathrm{des}}(t) & 0_{T}^{\mathrm{des}}(t) \\ \mathbf{0}^{\top} & 1 \end{bmatrix} $$

From §3.3.1, define the pose error resolved in $\{0\}$:

$$ e_p = {}^{0}p_{T}^{\mathrm{des}} - {}^{0}p_{T}, \qquad e_{\omega} = \log\left({}^{0}R_{T}^{\mathrm{des}}\, {}^{0}R_{T}^{\top} \right) $$

Attraction to the (first moving then fixed) target is a proportional twist with component-wise caps:

$$ v_{\mathrm{att}} = \mathrm{sat}_{v_{\max}}\left(K_p e_p\right), \qquad \omega_{\mathrm{att}} = \mathrm{sat}_{\omega_{\max}}(K_{\omega} e_{\omega}), $$

where $K_p, K_{\omega} \succeq 0$ are diagonal gains, and sat applies per-axis clamping to the translational and angular rates as in §3.3.3.

LSPB commands are jerk-bounded and pause/resume-ready; on RESUME, the time law continues from a consistent state to prevent discontinuities in $\dot{x}$ task.

Repulsion is driven by the minimum distance $d_{\min}$ from the TCP/link proxies to the human capsules (declared in §3.5, instantiated in Chapter 4), together with a unit direction $\hat{n}$ pointing from the nearest human proxy toward the TCP (all in $\{0\}$).

A smooth shaping $\phi(d)$ increases as distance shrinks (zero beyond a comfort band), e.g. logistic or reciprocal with hard caps. The repulsive twist is purely translational:

$$v_{\text{rep}} = k_r \phi(d_{\min}) \hat{n}, \qquad \omega_{\text{rep}} = \mathbf{0},$$

with $k_r > 0$ the repulsion gain and $\phi(d) = 0$ for $d \geq d_{\text{free}}$, $\phi'(d) \leq 0$, and $\phi(d)$ saturated at $v_{\max}$ to preserve boundedness.

Supervisor blending enforces STOP/RELEASE hysteresis. Let $w_{\text{att}}, w_{\text{rep}} \in [0,1]$ be state-dependent weights:

$$(w_{\text{att}}, w_{\text{rep}}) =$$

$$\begin{cases} (0,1) & \text{if } d_{\min} \leq \text{RIF\_STOP (HOLD/REPEL)}, \\ (\alpha, 1-\alpha) & \text{if RIF\_STOP} < d_{\min} < \text{RIF\_RELEASE}, \alpha \in (0,1), \\ (1,0) & \text{if } d_{\min} \geq \text{RIF\_RELEASE (RESUME)}. \end{cases}$$

The instantaneous TCP twist command in {0} is

$$\dot{x}_{\text{field}} = \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} w_{\text{att}}\, v_{\text{att}} + w_{\text{rep}}\, v_{\text{rep}} \\ w_{\text{att}}\, \omega_{\text{att}} \end{bmatrix}$$

then clamped by the global caps $(v_{\max}, \omega_{\max})$ from §3.3.3:

$$\dot{x}_{\text{task}} = \text{sat}_{(v_{\max}, \omega_{\max})}(\dot{x}_{\text{field}})$$

Supervisor hysteresis blends attraction and repulsion via $(\omega_{att}, \omega_{rep})$; stop and release radii enforce non-chattering boundary behavior (see §3.5.3)

Discretization and mapping to joints follow the same kinematic stack used elsewhere. At each control period Δt, the commanded joint rates are

$$\dot{q} = J_\lambda^{\#}(q)\dot{x}_{\text{task}} + N(q)\dot{q}_{\text{bias}}$$

with $J_\lambda^{\#}$ the SVD-based damped pseudoinverse (adaptive λ per §3.3.3), $N = I - J_\lambda^{\#} J$ the null-space projector, and $\dot{q}_{\text{bias}}$ a small posture/limit bias (§3.3.5). Joint-rate saturation (uniform scaling) is then applied to respect $|\dot{q}_i| \leq |\dot{q}|_{\max,i}$. Because $\dot{x}_{\text{task}}$ depends only on the instantaneous error and distance cues, the same law seamlessly (i) tracks a moving ${}^0T_T^{\text{des}}(t)$ with no re-planning, (ii) produces decisive retreat when $d_{\min}$ enters the STOP band, and (iii) continues toward the fixed goal once

RELEASE is met. The null-space term preserves the tool behavior while reshaping posture; leakage $\|JN\dot{q}_{\text{bias}}\|$ is monitored and clipped under the threshold defined in §3.3.5, ensuring the primary TCP objective remains pristine during approach, repel, and resume.

3.4.2 LSPB time law for the TCP (pause/resume–ready)

To impose predictable timing and smooth rates on the TCP, we drive both translation and orientation with a single scalar time law $s(t) \in [0,1]$ following a linear-segment with parabolic blends (LSPB). The reference pose evolves as

$$ {}^0p_T^{\text{des}}(t) = {}^0p_T^0 + s(t)\Delta p, \qquad {}^0R_T^{\text{des}}(t) = R_0\text{Exp}(s(t)\theta\hat{u}), $$

where $\Delta p = {}^0p_T^f - {}^0p_T^0$, and $R_0^\top R_f = \text{Exp}(\theta\hat{u})$ is the axis-angle gap between start and goal (rightinvariant, $\theta \in [0,\pi]$). Thus, the same normalized progress $s(t)$ synchronizes linear and angular motion.

Profile construction (continuous time)

Given path length $D = \|\Delta p\|$ and angle θ, we set bounds

$$ v_{\max}^{\text{lin}}, \qquad a_{\max}^{\text{lin}}, \qquad \omega_{\max}, \qquad \alpha_{\max} $$

and compute the progress-space limits that satisfy both translation and rotation:

$$ \dot{s}_{\max} = \min\left(\frac{v_{\max}^{\text{lin}}}{D+\varepsilon}, \frac{\omega_{\max}}{\theta+\varepsilon}\right), \qquad \ddot{s}_{\max} = \min\left(\frac{a_{\max}^{\text{lin}}}{D+\varepsilon}, \frac{\alpha_{\max}}{\theta+\varepsilon}\right), $$

with ε a tiny guard when D or θ is near zero. The LSPB has three phases: accelerate with $\ddot{s} = +\ddot{s}_{\max}$, cruise with $\dot{s} = \dot{s}_{\max}$, and decelerate with $\ddot{s} = -\ddot{s}_{\max}$. Let

$$ t_a = \frac{\dot{s}_{\max}}{\ddot{s}_{\max}}, \qquad s_a = \frac{1}{2}\ddot{s}_{\max}t_a^2 = \frac{1}{2}\frac{\dot{s}_{\max}^2}{\ddot{s}_{\max}} $$

be the progress covered during acceleration. If $2s_a \le 1$, there is a cruise phase with duration

$$ t_c = \frac{1-2s_a}{\dot{s}_{\max}}, \qquad T = t_a + t_c + t_a $$

Otherwise, the profile is triangular (no cruise). Set

$$\dot{s}_{\text{peak}} = \sqrt{\ddot{s}_{\max}}, \qquad t_a = \frac{\dot{s}_{\text{peak}}}{\ddot{s}_{\max}} = \frac{1}{\sqrt{\ddot{s}_{\max}}}, \qquad T = 2t_a, t_c = 0,$$

after normalizing the total progress to 1, The resulting piecewise law is

$$\dot{s}(t) = \begin{cases} \ddot{s}_{\max} t, & 0 \le t < t_a, \\ \dot{s}_{\max}, & t_a \le t < t_a + t_c, \\ \ddot{s}_{\max}(T - t), & t_a + t_c \le t \le T, \end{cases} \quad s(t) = \int_0^t \dot{s}(\tau) d\tau, s(0) = 0, s(T) = 1$$

Discrete-time realization (controller period Δt)

At each tick k :

$$\dot{s}_{k+1} = \text{clip}(\dot{s}_k + \Delta t \ddot{s}_k, 0, \dot{s}_{\max}), \qquad s_{k+1} = \text{clip}(s_k + \Delta t \dot{s}_{k+1}, 0,1),$$

with $\ddot{s}_k \in \{+\ddot{s}_{\max}, 0, -\ddot{s}_{\max}\}$ chosen by the phase scheduler. To bound discrete jerk, we limit changes of $\ddot{s}_k$:

$$|\ddot{s}_{k+1} - \ddot{s}_k| \le j_{\max} \Delta t$$

So; accelerations ramp between $\pm\ddot{s}_{\max}$ over a few ticks rather than switching instantaneously. The commanded twist sent to IK (resolved in {0} unless stated) is

$$v_k^{\text{des}} = \dot{s}_k \frac{\Delta p}{D + \varepsilon}, \qquad \omega_k^{\text{des}} = \dot{s}_k \theta \hat{u}$$

with component-wise clamps ensuring $\left\| v_k^{\text{des}} \right\| \le v_{\max}^{\text{lin}}$ and $\left\| \omega_k^{\text{des}} \right\| \le \omega_{\max}$. This yields the task demand $\dot{x}_{\text{task},k} = [v_k^{\text{des}\top} \quad \omega_k^{\text{des}\top}]^\top$ used in §3.3.

Axis synchronization

A single $s(t)$ guarantees that linear and angular segments (accelerate/cruise/decelerate) start and finish together. The limits are selected by the most restrictive of translational and rotational bounds, so timing is consistent and predictable even when θ/D varies across tasks.

Pause/resume semantics for SSM

When the supervisor asserts STOP (entry into the stop band), we freeze the time law by setting $\ddot{s}_k \to -\ddot{s}_{\max}$ until $\dot{s}_k \to 0$, then hold $\dot{s}_k = 0$ and keep s_k constant;

the commanded twist goes smoothly to zero without overshoot. On RELEASE (exit from the band and dwell satisfied), we resume from the stored $(s_k, \dot{s}_k = 0)$ and rebuild the remaining LSPB with the current bounds, preserving continuity of s and $\dot{s}$ and keeping jerk within the discrete limit above. If the goal pose is updated during a pause, we recompute $\Delta p, \theta, \hat{u}$ using the current TCP pose as the new start and continue from the same s_k (thereby avoiding discontinuities).

Why this serves our cell

The LSPB grants (i) reproducible arrival times and segment durations, (ii) synchronized translation-rotation with shared progress, (iii) bounded velocities/accelerations and discrete-time jerk, and (iv) clean pause/resume that interacts transparently with the supervisor. These properties make timelines and latency analyses in Chapters 5–6 interpretable, and they align with the safety and transparency requirements of collaborative operation.

3.4.3 Constraint enforcement: caps, saturation, and runtime monitors

This section formalizes how the TCP time-law (§3.4.1–§3.4.2) is executed safely in discrete time. The goal is to ensure that commanded task twists and the resulting joint motions remain within certified envelopes at every control tick, while preserving the timing semantics of each mode (notably pause/resume in Scenario 4).

Task-space limits and scaling

Let the nominal task twist be $\dot{x}_k^{\text{nom}} = [v_k^{\text{nom}\ \top} \quad \omega_k^{\text{nom}\ \top}]^\top$. We first enforce Euclidean-norm bounds with direction-preserving gains

$$\gamma_v = \min\left(1, \frac{v_{\max}^{\ln}}{\left\|v_k^{\min\|}\right\|\varepsilon}\right), \qquad \gamma_\omega = \min\left(1, \frac{\omega_{\max}}{\left\|\omega_k^{\max\|}\right\|\varepsilon}\right),$$

and set $v_k^{\text{cap}} = \gamma_v v_k^{\text{nom}}, \omega_k^{\text{cap}} = \gamma_\omega \omega_k^{\text{nom}}$. If per-axis limits apply, we additionally apply component-wise clipping:

$$[v_k^{\text{sat}}]_i = \text{clip}\left(\left[v_k^{\text{cap}}\right]_i, -v_{\max}^{(i)}, v_{\max}^{(i)}\right),$$

(and analogously for ω). These caps operate before IK so the solver receives physically realizable demands.

Discrete acceleration and jerk limiting

To bound transients independently of the time-law, a rate limiter constrains the increment from one tick to the next:

$$\dot{x}_k^{\mathrm{lim}} = \dot{x}_{k-1}^{\mathrm{lim}} + \mathrm{clip}\big(\dot{x}_k^{\mathrm{sat}} - \dot{x}_{k-1}^{\mathrm{lim}}, -a_{\max}^{x}\Delta t, a_{\max}^{x}\Delta t\big).$$

An optional slope-of-slope limiter bounds discrete jerk via $\left\|\ddot{x}_k^{\mathrm{lim}} - \ddot{x}_{k-1}^{\mathrm{lim}}\right\| \le j_{\max}^{x}$, with $\ddot{x}_k^{\mathrm{lim}} = \big(\dot{x}_k^{\mathrm{lim}} - \dot{x}_{k-1}^{\mathrm{lim}}\big)/\Delta t$. These bounds are configured tighter than ($\ddot{s}_{\max}, j_{\max}$) from the LSPB profile so emergency decelerations remain smooth.

Joint-space feasibility

Using the DLS IK of §3.3

$$\dot{q}_k^{\mathrm{nom}} = J_k^{\#}\dot{x}_k^{\mathrm{lim}} + N_k\dot{q}_k^{\mathrm{ns}},$$

we enforce joint-velocity limits via uniform scaling followed by per-joint clipping:

$$\beta = \min_i \min\left(1, \frac{\dot{q}_{\mathrm{m}}^{(I)}}{\left|\dot{q}_k^{\max}\right| \cdot | + \varepsilon}\right), \qquad \dot{q}_k^{\mathrm{cmd}} = \mathrm{clip}(\beta\dot{q}_k^{\mathrm{nom}}, -\dot{q}_{\max}, \dot{q}_{\max})$$

Position limits are handled by biasing the null-space command $\dot{q}_k^{\mathrm{ns}}$ away from bounds (barrier or quadratic wells), with the velocity caps guaranteeing instantaneous safety when bias is insufficient. Thresholds are declared in §3.5 and validated in §3.6.

Command validity and hold-last-safe

If any of the following occurs at tick k-IK failure or excessive $\kappa(J_k)$, stale/invalid target timestamps beyond LAT_THR, or unattainable acceleration/jerk-the controller transitions to a hold-last-safe policy:

$$\dot{q}_k^{\mathrm{cmd}} \leftarrow \mathrm{rate\_limit}\big(\mathbf{0}, \dot{q}_{k-1}^{\mathrm{cmd}}, a_{\max}^{q}\Delta t\big),$$

ensuring a smooth deceleration to rest while maintaining stability.

Runtime monitors and health flags

Lightweight runtime monitors execute each tick:

- **Timing monitor:** if a control-loop overrun exceeds TICK_THR, set MON_TICK=1 and enter hold-last-safe.
- **Velocity monitor:** if any $|\dot{q}_i| > \dot{q}_{\max}^{(i)} + \delta$, set MON_QDOT $= 1$, hard-clip, and log the event.
- **Target-stream monitor:** if target age > STREAM_THR, set MON_STREAM $= 1$, freeze the time-law $s(t)$ (pause semantics), and hold.
- **Saturation persistence counter:** if $\beta < 1$ or any component clip persists for N_{sat} ticks, raise SAT_PERSIST for later analysis (§6).

Interaction between sceanrios

- **Scenario 1 (vector attractive TCP, no human):** the proportional twist to a moving target (§3.4.1) is bounded by the task-space caps and rate limiters above; the IK then enforces joint feasibility.
- **Scenario 2 (scenario 1 + repulsive field):** identical timing and capping as Base 1, but the task twist is modulated by a repulsive component derived from human-proximity distances (introduced in Ch. 4). Constraint enforcement remains identical; only the input $\dot{x}_k^{\text{nom}}$ differs.
- **Scenario 3 (LSPB TCP, no human):** the LSPB time-law (§3.4.2) already shapes $\dot{x}$ with bounded accel/jerk; our limiters ensure feasibility under goal updates.
- **Scenario 4 (LSPB + SSM pause/resume):** same as scenario 3, with the target-stream monitor implementing STOP/RELEASE semantics by freezing or resuming the phase $s(t)$.
- **Scenario 5 (fixed-TCP null-space avoidance):** joint caps apply after null-space projection. If enforcing joint limits would otherwise corrupt the TCP task, we first scale $\dot{x}$ uniformly (task-space scaling) to preserve task integrity.

3.5 Safety variables and thresholds (declared here, used later)

This section declares the global safety variables, units, and semantics used throughout Chapters 4–6. They serve two purposes: (i) to make the control logic in §3.4 unambiguous, and (ii) to ensure every experiment logs comparable, audit-ready signals.

3.5.1 Coordinate conventions and units

All distances and positions are expressed in meters in the world frame $\{0\}$. Rotations are parameterized either by axis–angle (radians) or quaternion (unit norm); angular errors are reported as minimal-angle magnitudes in radians unless stated otherwise. Rates are per-second; timestamps are UTC with millisecond resolution.

3.5.2 Core tolerances

- **POS_TOL [m]:** maximum admissible Euclidean position error at the TCP before a position-converged flag is raised. Used by: stop/resume checks (§4.4), success criteria (§6).
- **ROT_TOL [rad]:** maximum admissible orientation error (angle of $R_{\text{des}}\,R^{\top}$). Entering/leaving the orientation-hold band is determined by this threshold.
- **VEL_TOL** $[\mathbf{m/s}]$, $[\mathbf{rad/s}]$**:** small-band threshold below which the TCP is treated as stationary for state transitions.

3.5.3 Repulsion and SSM thresholds

Let d denote the minimum distance between any human capsule and any robot proxy (defined in Ch. 4).

- **RIF_STOP** $[\mathbf{m}]$**:** enter-stop threshold. When $d \leq$ RIF_STOP, the supervisor forces STOP (scenario 4) or maximum repulsion (scenario 2), regardless of attractive commands.
- **RIF_RELEASE [m]:** exit-stop threshold. Normal operation resumes only when $d \geq$ RIF_RELEASE and the dwell condition holds (§4.4). Hysteresis requires RIF_RELEASE > RIF_STOP.

- **DWELL_SSM [s]:** minimum time the system must remain continuously outside the stop band before RELEASE, to avoid chatter.
- **RIF_GAIN_MAX [-]:** upper bound on the repulsive-field gain used to shape the task twist in scenario 2; ensures bounded commanded velocities.

3.5.4 Fixed-TCP avoidance and leakage

For scenario 5 (null-space avoidance with a fixed TCP):

- **LEAK_THR [m/s], [rad/s]:** maximum admissible task-space leakage induced by joint-limit handling or null-space injections, quantified as $\|\Delta \dot{x}_{\mathrm{TCP}}\|$ between pre/post projection. If exceeded, uniform taskspace scaling is applied (§4.5), and an event is logged.
- **NS_GAIN_MAX [-]:** bound on the null-space step to keep avoidance smooth and secondary to the primary task.
- **LAT_THR [s]:** maximum allowed age of the target/pose stream; if exceeded, the time law is frozen (scenario 4 pause semantics) and hold-last-safe is engaged (§3.4.3).
- **TICK_THR [s]:** maximum allowed control-loop overrun before triggering a timing fault and smooth deceleration to rest. Caps are applied before inversion; joint-space limits are enforced uniformly across modes to keep behavior consistent.
- $\boldsymbol{v}_{\mathbf{max}}^{\mathbf{lin}}$, $\boldsymbol{\omega}_{\mathbf{max}}$ $[\mathbf{m/s}], [\mathbf{rad/s}]$ **:** task-space speed caps used in §3.4.3.
- $\boldsymbol{a}_{\mathbf{max}}^{\boldsymbol{x}}$, $\boldsymbol{j}_{\mathbf{max}}^{\boldsymbol{x}}$ $[\ \mathbf{m/s^2}], [\mathbf{m/s^3}]$ **and rotational counterparts**: task-space acceleration/jerk caps (independent of LSPB).
- $\dot{\boldsymbol{q}}_{\mathbf{max}}$, $\boldsymbol{a}_{\mathbf{max}}^{\boldsymbol{q}}$ $[\mathbf{rad/s}], [\mathbf{rad/s^2}]$ **:** joint-space limits used uniformly across modes.

3.5.5 Health flags and logging schema

At every control tick the following health flags are evaluated and logged alongside raw signals:

- MON_TICK (timing overrun > TICK_THR)

- MON_STREAM (target age > LAT_THR)
- IK_FAIL (solver failure or $\kappa(J)$ over limit)
- SAT_PERSIST (velocity/acceleration saturation sustained $> N_{\text{sat}}$ ticks)
- LEAK_EVT (task leakage > LEAK_THR in scenario 5)
- RESUME_OK (pause/resume cycle completed without chattering at thresholds)
- SSM_STATE ∈ { Approach, Hold, Repel, Resume, Stop } (Scenarios 2 and 5, §4.4)

Logs include: (i) target/TCP poses and errors, (ii) task twists pre/post capping, (iii) joint velocities/torques, (iv) min distances and nearest-pair IDs, (V) mode/state variables, (vi) timestamps and loop periods. Files are stored per run with immutable metadata: seed, configuration hash, software versions, and scene manifest (§5.4).

3.5.6 Defaults and calibration pointers

Nominal default values are provided as starting points and are refined in §3.6.3 via a calibration sweep. Hysteresis pairs (RIF_STOP, RIF_RELEASE) are set from capsule radii and sensing noise; tolerances (POS_TOL, ROT_TOL) reflect controller accuracy at steady state; latency guard LAT_THR derives from the end-to-end budget in §3.6.2.

3.6 Identification and validation of the model

This section establishes how the kinematic model and its use in the controller are verified before any experimental runs are accepted. The objective is to demonstrate, with traceable evidence, that (i) the analytic Jacobian implemented in the stack matches the scene-consistent forward kinematics to within tight numerical tolerances, (ii) the end-to-end timing of sensing, decision, and actuation respects the controller period with quantified latency and jitter, (iii) the tolerances and thresholds declared in Chapter 3 are calibrated against observed behavior rather than chosen ad hoc, and (iv) every result is reproducible from versioned configurations and logged artifacts.

The validation proceeds along four axes. First, Jacobian correctness is checked by unit tests that compare analytic columns against finite-difference estimates of the forward kinematics at randomized joint configurations spanning the feasible set; pass/fail thresholds are defined a priori in both translational and rotational components. Second, a timing and latency budget is measured under the same synchronous stepping used in experiments: the pipeline from distance updates and state acquisition to command emission is instrumented, the distribution of latencies is reported, and an overrun policy is enforced whenever the measured delay approaches the controller period. Third, the operational tolerances and safety thresholds introduced earlier—position and orientation bands for the TCP, separation hysteresis for stop and release, and the leakage bound for null-space actions—are calibrated by sweeps that trade tracking performance against safety margins; the chosen values are those that satisfy the pass criteria while preserving transparency of motion. Fourth, reproducibility is guaranteed by storing configuration files, seeds, and version hashes together with run manifests, so that any table or figure can be regenerated exactly.

3.6.1 Finite-difference vs. analytic; unit tests; pass/fail

Before any experiment is admitted, the geometric Jacobian implemented in the stack is verified against finite-difference estimates of the forward kinematics generated from the same, scene-consistent link frames (§3.2). The goal is to prove that each analytic column correctly maps an infinitesimal change in the corresponding joint to the induced instantaneous linear and angular velocity at the TCP, under the declared frame resolution.

Protocol and test set

We validate on a fixed-size batch of 500 joint configurations drawn uniformly within the conservative joint limits (§3.2), with a 5° margin from each bound to avoid hard-limit artifacts. A deterministic seed fixes the sample set for reproducibility. All quantities are resolved in the base/world frame {0}; a mirrored run repeats the checks in the tool frame {T} using the rigid rotation described in §3.3.2, and the two outcomes are cross-checked for consistency.

Finite-difference model (ground truth)

For each configuration q and each joint index j ∈ {1, …,7}, we evaluate the forward kinematics at q ± he, j with a central step h = 10^{-6}rad(ej is the j -th basis vector). The translational column "truth" is formed as

$$\hat{J}_{v,-j} = \frac{{}^{0}p_T(q + he_j) - {}^{0}p_T(q - he_j)}{2h} \in \mathbb{R}^3$$

and the rotational column "truth" uses the right-invariant orientation increment via the matrix logarithm:

$$\hat{J}_{\omega,-j} = \frac{\log\left({}^{0}R_T(q)^{\top}\,{}^{0}R_T(q + he_j)\right) - \log\left({}^{0}R_T(q)^{\top}\,{}^{0}R_T(q - he_j)\right)}{2h} \in \mathbb{R}^3.$$

This yields a numerically robust estimate of the instantaneous twist induced by joint j, directly comparable to the analytic column $J_{.j} = \left[J_{v,.j}^{\top} J_{\omega,.j}^{\top}\right]^{\top}$.

Error metrics and aggregation

For each (q_1, j) we compute absolute and relative errors

$$e_{v,j}^{\text{abs}} = \left\|J_{v,j} - \hat{J}_{v,j}\right\|_2, \qquad e_{\omega,j}^{\text{abs}} = \left\|J_{\omega,j} - \hat{J}_{\omega,j}\right\|_2,$$

$$e_{v,j}^{\text{rel}} = \frac{e_{v,j}^{\text{abs}}}{\max\left(\left\|\hat{J}_{v,j}\right\|_2, \varepsilon_v\right)}, \qquad e_{\omega,j}^{\text{rel}} = \frac{e_{\omega,j}^{\text{abs}}}{\max\left(\left\|\hat{J}_{\omega,j}\right\|_2, \varepsilon_\omega\right)},$$

with $\varepsilon_v = 10^{-9}$ and $\varepsilon_\omega = 10^{-9}$ to avoid division by very small denominators. We report, per run: (i) the maxima over all columns ("worst column"), (ii) the 95th percentiles (robust spread), and (iii) means (central tendency). In parallel, the Jacobian's condition number $\kappa(J)$ is recorded to contextualize errors near singular neighborhoods.

Pass/fail thresholds (applied deterministically)

A validation run passes if all the following are satisfied simultaneously:

- **Worst-column absolute errors:** $\max_{q,j} e_{v,j}^{\text{abs}} \le 5 \times 10^{-6}$ m and $\max_{q,j} e_{\omega,j}^{\text{abs}} \le 5 \times 10^{-6}$rad.

- **Robust relative errors:** 95th percentiles satisfy $e_{v,j}^{\mathrm{rel}} \le 1 \times 10^{-4}$ and $e_{\omega,j}^{\mathrm{rel}} \le 1 \times 10^{-4}$.
- **Frame-consistency check:** for every q,

$$\left\| \mathrm{blkdiag}(R_T^{\top}, R_T^{\top}) J_{\{0\}}(q) - J_{\{T\}}(q) \right\|_F \le 10^{-8},$$

ensuring the base-resolved and tool-resolved Jacobians are rigidly consistent.

- **SVD projector check**: the projector $N = I_7 - J_\lambda^{\#} J$ (with the same DLS settings used in control) satisfies $\|JN\|_F \le 10^{-8}$, confirming that the null-space action is orthogonal to the primary task numerically.

Implementation notes (traceability)

All quantities are computed with the same forward-kinematics function used in the controller, ensuring that the comparison isolates the Jacobian construction rather than mixing models. The step h is small enough to capture the local derivative while remaining above machine epsilon for the scale of the scene; we log $(h, \varepsilon_v, \varepsilon_\omega)$, the random seed, joint limits, and the full set of maxima/percentiles so that the table of validation metrics (Table 3.5) can be regenerated exactly. Any configuration with $\kappa(J)$ exceeding 10^8 is still included; large relative errors in such cases are expected and are discussed separately in the identification notes, but the absolute error criteria above remain the formal pass conditions.

Outcome artifacts

The unit-test harness emits: (i) a CSV of per-sample, per-joint errors $\left(e_{v,j}^{\mathrm{abs}}, e_{\omega,j}^{\mathrm{abs}}, e_{v,j}^{\mathrm{rel}}, e_{\omega,j}^{\mathrm{rel}}, \kappa(J)\right)$; (ii) a summary row with worst, 95th-percentile, and mean values; (iii) the frame-consistency residuals and projector residuals; and (iv) a pass/fail flag. These are the inputs for Table 3.5 (validation metrics & pass criteria) referenced at the end of §3.6.

3.6.2 Timing and latency budget (controller tick vs. physics step; end-to-end latency; overrun policy)

Reliable interpretation of results requires that sensing, control, and actuation advance with a known cadence and bounded delay. This subsection fixes the timing model used across all experiments, the method used to measure end-to-end latency,

the acceptance thresholds, and the policy applied whenever a cycle risks overrunning its budget.

Timing model and notation

The controller runs with a fixed period T_c (tick rate $f_c = 1/T_c$). The simulator's physics integrator advances with step T_p (rate $f_p = 1/T_p$). We operate the scene in synchronous mode with T_p chosen as a submultiple of T_c (default $T_c = 5\ \text{ms}, T_p = 1\ \text{ms}$), so one control tick wraps five physics substeps. Within a tick, the end-to-end delay from sensing to actuation is decomposed as

$$L_{\text{e 2 e}} = L_{\text{sense}} + L_{\text{queue}} + L_{\text{comp}} + L_{\text{commit}},$$

where L_{sense} is the time between the physics state at the sampling instant and stamped availability of joint/pose signals, L_{queue} is any middleware/buffer delay, L_{comp} is the wall-time for the control computation (FK/Jacobian, SVD/DLS, references, supervision), and L_{commit} is the time to deliver the command to the simulator's actuator at the next integrator boundary. We track loop jitter as $\Delta T_c(k) = T_c^{\text{actual}}(k) - T_c$.

Measurement procedure

All timing is measured on a single monotonic clock used by the synchronous stepping loop. Each control cycle logs: (i) tick start time t_k; (ii) timestamp of the physics state sampled $\hat{t}_k$; (iii) computation start/stop: (iv) command commit time. From these we compute per-cycle $L_{\text{sense}}, L_{\text{queue}}, L_{\text{comp}}, L_{\text{commit}}, L_{\text{e2e}}$ and ΔT_c. A mirrored run is recorded with the Jacobian resolved in $\{T\}$ to confirm resolution choice has no timing side-effects (it does not change timings by design). The emitted artifact (for Fig. 3.4) is a histogram of L_{e2e} with overlays for mean, median, p_{95}, p_{99}, and maximum, plus a separate plot of ΔT_c over time to visualize burstiness.

Budget and acceptance thresholds (applied deterministically)

We partition the control period into a hard budget for computation and a soft budget for I/O:

$$L_{\text{comp}} \le \beta T_c, \qquad L_{\text{sense}} + L_{\text{queue}} + L_{\text{commit}} \le (1-\beta)T_c$$

with $\beta = 0.6$ by default. A run is admitted if all hold:

- Mean end-to-end delay $\bar{L}_{\mathrm{e2e}} \leq 0.7T_{c:}$
- $p_{95}(L_{\mathrm{e2e}}) \leq 0.85T_c$ and $p_{99}(L_{\mathrm{e2e}}) \leq 0.95T_{c;}$
- Worst-case jitter $|\Delta T_c|_{\max} \leq 0.2$ ms (for $T_c = 5$ ms); RMS jitter $\leq$ 0.05 ms;
- Overrun rate (cycles with $L_{\mathrm{e2e}} > T_c$) equals 0 over the validation window; if nonzero during development sweeps, it must be $< 10^{-4}$ and cannot cluster (no more than one overrun in any 1-s window).

Overrun detection and hold-last-safe policy

An overrun is declared at tick k if $t_k + L_{e2e} \geq t_k + T_c$. In that case the system:

1. freezes the commanded twist and joint rates for the upcoming tick (hold-last-safe);
2. stamps a health flag (OVERRUN=1) and increments a counter;
3. drops any queued intermediate sensor updates to realign sampling to the next physics boundary:
4. reduces internal task gains by a factor $\eta \in (0,1)$ (default $\eta = 0.7$) for the next tick to avoid a second consecutive overrun.

If two consecutive overruns occur, the supervisor asserts a PAUSE, which zeroes the task demand, preserves the last valid posture, and resumes only after a clean tick with $L_{e2e} < 0.7T_c$. All events are logged with cycle indices for post-hoc traceability.

Synchronization choices and drift control

Because the loop is simulator-paced, drift between controller and physics time is structurally prevented: the next control tick cannot start until the physics step acknowledges the prior commit. To forestall numeric drift in the down-counter, the scheduler re-anchors to the simulator epoch every 100 ticks (configurable) and logs the re-anchor residual (target < 20μ s).

Notes on interplay with controllers

The timing budget is identical across vector-field and LSPB reference generators; LSPB adds a negligible lookup/interpolation cost absorbed in L_{comp}. Null-space computations reuse the same SVD already computed for DLS, so they do not change the asymptotic cost; their presence is nevertheless recorded in the log header for completeness.

3.6.3 Tolerance and threshold calibration (POS/ROT tolerances; STOP/RELEASE bands; leak threshold protocol)

This subsection fixes how tracking tolerances and supervisory thresholds are selected so that all later experiments are executed under declared margins. The goal is to set values that are tight enough to be informative and reproducible, yet conservative enough to avoid spurious interventions.

Scope and notation

We calibrate: (i) the TCP tracking tolerances POS_TOL (meters) and ROT_TOL (radians) used to judge regulation; (ii) the STOP/RELEASE separation bands (RIF_STOP, RIF_RELEASE) used by the supervisor; and (iii) the null-space leak threshold LEAK_THR that limits corruption of the primary task. All distances are world-frame, all orientation errors use the axis-angle norm from §3.3.1 , and leakage is measured as $\|JN\dot{q}_{\text{bias}}\|$ using the same Jacobian resolution chosen for control (recorded in the log header).

Calibration procedure-TCP tolerances

- **Static posture hold (noise floor).** With the robot immobilized in the simulator (no reference motion), we log 30 s of TCP pose to estimate the sensor/quantization floor: σ_p(m) and σ_ω(rad). We require these floors to be at least an order of magnitude below the eventual tolerances.
- **Ramp-in regulation (closed-loop capability).** We command exponentially decaying references to the current pose and measure steady-state residuals $\bar{e}_p, \bar{e}_\omega$ after transients (last 10 s).

- **Tolerance selection.** We set POS_TOL $= k_p \max(\sigma_p, \bar{e}_p)$ and ROT_TOL $= k_\omega \max(\sigma_\omega, \bar{e}_\omega)$ with $k_p = k_\omega = 5$ by default. These multipliers ensure false positives are rare while keeping bounds informative.
- **Validation.** We replay nominal trajectories (vector and LSPB) and verify that at least 99% of samples satisfy $\|e_p\| \leq$ POS_TOL and $\|e_\omega\| \leq$ ROT_TOL; violations trigger either gain retuning or tolerance reestimation.

Calibration procedure-STOP/RELEASE hysteresis

- **Distance trace acquisition.** With the human avatar approaching and receding along representative paths, we log the minimum robot-human distance $d_{\min}(t)$ (capsule-proxy model from Ch. 4).
- **Band placement.** We set RIF_STOP at the smallest distance for which the supervisor must pause to guarantee clearance under worst-case approach rates, and RIF_RELEASE > RIF_STOP to introduce hysteresis. Practically, we sweep candidate pairs over a grid and simulate approach-hold-resume episodes; for each pair we measure stop latency, minimum achieved clearance, and chatter events.
- **Acceptance.** Choose the smallest RIF_STOP that yields a measured minimum clearance $\geq d_{\text{req}}$ (declared in Ch. 4) with zero chatter, and the smallest RIF_RELEASE that guarantees resume only after the clearance has exceeded RIF_STOP by at least Δd_{hyst} (default 0.05 m). The selected pair is fixed for all experiments of the same sensing fidelity and controller period.

Calibration procedure-null-space leak threshold

- **Baseline measurement.** With the primary task active and a neutral bias $\dot{q}_{\text{bias}} = 0$, we measure $\|JN\dot{q}_{\text{bias}}\|$ to characterize numerical leakage (should be ~ 0 within solver precision).
- **Bias injection sweeps.** We inject bounded biases representative of posture shifts and repulsion fields (magnitudes spaced logarithmically), compute

the resulting leakage, and record how much of the bias penetrates the primary task.

- **Threshold choice.** We set LEAK_THR to the largest value that preserves the primary task within tolerances, i.e., the smallest threshold for which the induced task error remains $\leq$ POS_TOL/ROT_TOL over all sweeps. In practice, LEAK_THR is chosen near the 95th percentile of observed leakage under maximum expected bias.

3.6.4 Reproducibility artifacts (configs, seeds, version hashes, run manifests)

To make every result in this chapter independently repeatable, we fix a concrete set of artifacts that capture the full provenance of each experiment. The intent is that a reader can re-execute any run and obtain numerically consistent traces (up to floating-point noise) by relying only on these artifacts. This subsection defines what is stored, how it is named, and how integrity is verified.

Scope and guiding principles

We record the exact code and scene versions used; the full configuration (including safety thresholds and time-law parameters); all random seeds; the execution environment; and the outputs with units and sampling rates. Artifacts are organized so that (i) one manifest describes one run end-to-end, (ii) content hashes guarantee immutability, and (iii) any non-determinism is bounded and disclosed.

3.7 Conclusions

This chapter established the kinematic foundation and supervisory scaffolding on which the remainder of the thesis is built. We began by motivating the use of a 6×7, velocity-level formulation for a 7-DoF Franka Emika Panda operating in a collaborative cell and by fixing the scene-consistent frames, limits, and link proxies that make simulation runs reproducible and transferable to hardware. Forward kinematics and a small-angle, axis–angle pose-error definition were formalized to avoid parametrization singularities while remaining well-conditioned for incremental corrections typical of supervised collaboration. The geometric Jacobian was derived directly from the simulator-aligned link frames, with explicit frame-resolution conventions and verification procedures to ensure agreement between analytic and finite-difference evaluations.

Building on these primitives, we specified the inverse-kinematics operator used throughout: a damped least-squares pseudoinverse constructed from the Jacobian's SVD, with adaptive damping tied to conditioning, bounded twist inputs, and uniform joint-rate saturations. We complemented these choices with manipulability and conditioning metrics that define safe neighborhoods and guide gain/damping schedules as the arm approaches singular regions or joint-limit boundaries. On top of the primary TCP task, we introduced the task-priority composition that preserves the tool objective while allocating the null space to posture shaping and later, human-aware safety behaviors. A quantitative leak guard was defined to certify that secondary actions do not corrupt the primary task. For scenarios requiring an unmoving tool, we formalized orientation locking so that fixed-TCP constraints can coexist with null-space reconfiguration.

Time-law generation for the TCP was then framed along two complementary paths. A continuous vector-attractive reference offers responsiveness and simplicity and serves as the baseline for approach/repel behaviors when tracking a moving target before converging to a fixed goal. In contrast, the LSPB scheme imposes an explicit accelerate–cruise–decelerate structure with axis synchronization, bounded jerk in discrete time, and well-defined pause/resume semantics that align with the supervisor's STOP/RELEASE logic. Constraint enforcement—velocity and acceleration caps, command saturation, and cycle-integrity monitors—was specified so that reference generation, inversion, and supervision operate within declared limits and remain diagnosable from logs.

Finally, we consolidated the safety variables and thresholds that recur across chapters (distances in the world frame, STOP/RELEASE hysteresis bands with dwell, tracking tolerances, and the leak threshold) and fixed the identification-and-validation procedures. These include Jacobian unit tests against finite differences, an end-to-end timing/latency budget with a hold-last-safe policy on overruns, tolerance/threshold calibration protocols, and a full set of reproducibility artifacts (manifests, seeds, version hashes, and run bundles) that anchor every reported figure and table.

Chapter 4

Human Model, Distances and Safety Behaviors

Chapter 4 formalizes the human–robot interaction layer used in the experiments. Starting from time-stamped human pose streams, we construct skeleton-derived capsules that serve as collision proxies, define reference-frame distances and a budgeted nearest-pair query, and build two safety behaviors on this foundation: continuous repulsive fields blended with the posture bias for smooth approach–repel transitions (Scenario 2), and a supervisor with explicit STOP/RELEASE hysteresis that pauses and resumes an LSPB TCP time law without corrupting its schedule (Scenario 4). The framework is then extended to fixed-TCP avoidance in redundant kinematics (Scenario 5), where the full 6-DoF task is preserved and avoidance acts in the null space.

All variables required by the subsequent chapters are declared here, including distance definitions, tolerances, STOP/RELEASE bands, and leak thresholds, together with logging flags for experiment health. Section 4.1 introduces the pose sources, filtering, and capsule layout; Section 4.2 specifies the distance computation and computational budget; Section 4.3 presents the repulsive field shaping and blending with the posture bias; Section 4.4 describes the finite-state supervisor and its timing guarantees; Section 4.5 develops fixed-TCP null-space avoidance and orientation locking; Section 4.6 discusses stability and transparency considerations.

4.1 Human pose streams to skeleton-derived capsules

This section formalizes how 3D joint streams are transformed into a compact proxy used by the safety modules and by the simulator mannequin.

4.1.1 Input and world alignment

Let the sensing frame be {K} and the global laboratory frame be {0}. At time t the sensor delivers joint positions

$$^{K}p_i(t) \in \mathbb{R}^3, \qquad i \in \mathcal{J}$$

for the set $\mathcal{J} = \{\text{HC}, \text{SC}, \text{Head}, \text{LS}, \text{LE}, \text{LW}, \text{RS}, \text{RE}, \text{RW}, ...\}$ (hip center, shoulder center, head, left/right shoulder, elbow, wrist, etc.). After causal filtering and gap filling, positions are mapped to {0} via a fixed rigid transform

$$^{0}p_i(t) = {}^{0}R_K \, {}^{K}p_i(t) + {}^{0}p_K,$$

with

$$^{0}R_K = R_z(-135^\circ), \qquad {}^{0}p_K \in \mathbb{R}^3,$$

chosen so that the hip line aligns with the table edge and the floor height is consistent with the robot scene. All subsequent computations use $p_i(t)$ in meters [11, 16].

4.1.2 Local anatomical frames and mannequin actuation

Denote right-side triplet (RS, RE, RW) and left-side triplet (LS, LE, LW). Unit directions are built from adjacent segments.

Torso frame about the shoulder center SC:

$$\hat{z}_{\text{torso}} = \frac{{}^{9}p_{\text{SC}} - {}^{9}p_{\text{HC}}}{\| {}^{0}p_{\text{SC}} - {}^{9}p_{\text{HC}} \|}, \qquad \hat{y}_{\text{torso}} = \frac{\hat{z}_{\text{torso}} \times ({}^{0}p_{\text{LS}} - {}^{9}p_{\text{RS}})}{\| \hat{z}_{\text{torso}} \times ({}^{0}p_{\text{LS}} - {}^{0}p_{\text{RS}}) \|},$$

$$\hat{x}_{\text{torso}} = \hat{y}_{\text{torso}} \times \hat{z}_{\text{torso}}$$

The corresponding rotation is

$$^{\text{torso}}R_0 = [\hat{x}_{\text{torso}} \quad \hat{y}_{\text{torso}} \quad \hat{z}_{\text{torso}}]^{\top}$$

Right shoulder frame at RS:

$$\hat{x}_{\mathrm{RS}} = \frac{{}^{0}p_{\mathrm{RS}} - {}^{0}p_{\mathrm{RE}}}{\| {}^{0}p_{\mathrm{RS}} - {}^{0}p_{\mathrm{RE}} \|}, \qquad \hat{z}_{\mathrm{RS}} = \frac{({}^{0}p_{\mathrm{RW}} - {}^{0}p_{\mathrm{RE}}) \times ({}^{0}p_{\mathrm{RS}} - {}^{0}p_{\mathrm{RE}})}{\| ({}^{0}p_{\mathrm{RW}} - {}^{0}p_{\mathrm{RE}}) \times ({}^{0}p_{\mathrm{RS}} - {}^{0}p_{\mathrm{RE}}) \|},$$

$$\hat{y}_{\mathrm{RS}} = \hat{z}_{\mathrm{RS}} \times \hat{x}_{\mathrm{RS}}, \qquad {}^{\mathrm{RS}}R_0 = [\hat{x}_{\mathrm{RS}} \hat{y}_{\mathrm{RS}} \hat{z}_{\mathrm{RS}}]^{\top}.$$

Left shoulder frame at LS (sign convention matches the mannequin):

$$\hat{x}_{\mathrm{LS}} = -\frac{{}^{0}p_{\mathrm{LE}} - {}^{0}p_{\mathrm{LS}}}{\| {}^{0}p_{\mathrm{LE}} - {}^{0}p_{\mathrm{LS}} \|}, \qquad \hat{z}_{\mathrm{LS}} = \frac{({}^{0}p_{\mathrm{LS}} - {}^{0}p_{\mathrm{LE}}) \times ({}^{0}p_{\mathrm{LW}} - {}^{0}p_{\mathrm{LE}})}{\| ({}^{0}p_{\mathrm{LS}} - {}^{0}p_{\mathrm{LE}}) \times ({}^{0}p_{\mathrm{LW}} - {}^{0}p_{\mathrm{LE}}) \|},$$

$$\hat{y}_{\mathrm{LS}} = \hat{z}_{\mathrm{LS}} \times \hat{x}_{\mathrm{LS}}, \qquad {}^{\mathrm{LS}}R_0 = [\hat{x}_{\mathrm{LS}} \hat{y}_{\mathrm{LS}} \hat{z}_{\mathrm{LS}}]^{\top}$$

Skeleton timestamps and frame validity flags are preserved end-to-end and used by the supervisor during gating (see §5.1).

Elbow flexion angles (for revolute elbows) follow from relative orientations. With forearm frames ${}^{\mathrm{RE}}R_0$ and ${}^{\mathrm{LE}}R_0$ built from the segments (RE → RW) and (LE → LW), the right-elbow rotation in the upper-arm frame is

$${}^{\mathrm{RS}}R_{\mathrm{RE}} = {}^{\mathrm{RS}}R_0 \, {}^{0}R_{\mathrm{RE}}, \qquad \theta_{\text{R-elbow}} = \mathrm{EA}_{123}({}^{\mathrm{RS}}R_{\mathrm{RE}})_3$$

and similarly for the left elbow, where $\mathrm{EA}_{123}(\cdot)_3$ extracts the third XYZ Euler angle used by the mannequin. Shoulder and torso spherical joints are commanded directly via the corresponding direction-cosine matrices ${}^{\mathrm{RS}}R_0$, ${}^{\mathrm{LS}}R_0$, and ${}^{\text{torso}}R_0$ flattened in column-major order.

4.1.3 Capsule proxy set

The skeleton is reduced to six convex proxies updated at the sensor rate:

$$\mathcal{C}(t) = \{(a_c(t), b_c(t), r_c) \mid c = 1, \ldots, 5\} \cup \{(h(t), r_H)\},$$

with endpoints

$$c = 1{:}\,(a_1, b_1) = ({}^{0}p_{\mathrm{RS}}, {}^{0}p_{\mathrm{RE}}), \qquad c = 2{:}\,(a_2, b_2) = ({}^{0}p_{\mathrm{RE}}, {}^{0}p_{\mathrm{RW}}),$$
$$c = 3{:}\,(a_3, b_3) = ({}^{0}p_{\mathrm{LS}}, {}^{0}p_{\mathrm{LE}}), \qquad c = 4{:}\,(a_4, b_4) = ({}^{0}p_{\mathrm{LE}}, {}^{0}p_{\mathrm{LW}}),$$
$$c = 5{:}\,(a_5, b_5) = ({}^{0}p_{\mathrm{Abd}}, {}^{0}p_{\mathrm{Spine}}), \qquad h(t) = {}^{0}p_{\mathrm{Head.}}$$

Radii $\{r_c\}$ are conservative constants that cover soft tissue, clothing, and residual pose noise. Default values used in experiments are reported in Table 4.1. This table lists per-segment radii in "meters" and is reused unchanged in Chapters 5–6.

Parameter	Symbol	Value	Unit
Max task speed (cruise cap)	v_max	1.2	m/s
Max task acceleration (braking cap)	a_max	1.2	m/s²
Acceptance / deadband	r_accept	0.05	m
Pause dwell (down)	T↓	0.25	s
Final hold duration	T_hold	2.0	s
Stop radius	r_stop	0.25	m
Release radius	r_release	0.28	m
Joint speed cap	‖q̇‖_∞	1.0	rad/s
Per-tick step cap	step_cap	6	deg/step
Linearization radius	d_lin	0.20	m
Damping factor	λ	—	–
Posture weight	w_post	—	–
Repulsion weight	w_rep	—	–
Joint-limit weight	w_lim	—	–
Manipulability weight	w_m	—	–
Joint-limit margin	Δq_lim	—	rad

Table 4.1: Velocity/acceleration caps, damping λ, acceptance radius, pause/resume dwell times, hysteresis bands (r_stop, r_release), posture weights (w_post), repulsion weights (w_rep), joint-limit margins, manipulability weight (w_m); SI units.

4.1.4 Signed distance to a capsule

For any query point $p \in \mathbb{R}^3$ and capsule (a, b, r),

$$\lambda^* = \text{clip}_{[0,1]}\left(\frac{(p-a)^\top (b-a)}{\|b-a\|^2}\right), \qquad \pi(p) = a + \lambda^*(b-a),$$

$$d_{\text{cap}}(p; a, b, r) = \|p - \pi(p)\| - r.$$

For a head sphere $(h, r_H), d_{\text{sph}}(p; h, r_H) = \|p - h\| - r_H$. The instantaneous human-proxy distance field is the minimum over the set,

$$d_{\text{human}}(p, t) = \min\left(\min_{c=1,\dots,5} d_{\text{cap}}(p; a_c(t), b_c(t), r_c), d_{\text{sph}}(p; h(t), r_H)\right).$$

4.1.5 Timing and coherence

All joint samples ${}^0p_i(t)$ carry time stamps. A hold-last-good policy provides a coherent snapshot $\{{}^0p_i(\hat{t})\}$ to the safety layer when a new frame is late, and frames flagged as unreliable by the front-end filter are not propagated downstream.

4.2 Clearance distances and minimum-distance query

4.2.1 Robot points of interest

Let the robot be instrumented with a finite set of witness points $\mathcal{P}_r = \{p_k\}_{k=1}^{N_r} \subset \mathbb{R}^3$ expressed in the world frame $\{0\}$. At time t,

$$p_k(t) = f_k(q(t)), k = 1, \dots, N_r$$

with $q \in \mathbb{R}^n$ the joint vector.

4.2.2 Human proxy set

The capsule set $\mathcal{C}(t)$ is defined in 4.1. For a capsule (a, b, r),

$$\lambda^\star(p; a, b) = \text{clip}_{[0,1]} \frac{(p-a)^\top (b-a)}{\|b-a\|^2}, \qquad \pi(p; a, b) = a + \lambda^\star(b-a)$$
$$d_{\text{cap}}(p; a, b, r) = \|p - \pi(p; a, b)\| - r$$

and for a sphere (h, r_H),

$$d_{\text{sph}}(p; h, r_H) = \|p - h\| - r_H$$

4.2.3 Effective radii and signed clearance

Optional padding for robot and human is modeled by

$$\tilde{d}(p; \mathcal{C}) = \min\left(\min_{c=1..5} d_{cap}(p; a_c, b_c, r_c + \rho_r), d_{sph}(p; h, r_H + \rho_r) \right)$$

with $\rho_r \geq 0$ the robot's protective radius. Setting $\rho_r = 0$ recovers geometric distances. Composite SDFs are a drop-in alternative when mesh fidelity is needed [9].

4.2.4 Global and groupwise minima

The instantaneous global clearance is

$$d_{\min}(t) = \min_{k=1..N_r} \tilde{d}(p_k^0(t); \mathcal{C}(t))$$

For downstream shaping, witness points are partitioned in ordered link groups $\{\mathcal{G}_g\}_{g=1}^{G}$ (proximal to distal). The per-group minima are

$$d_g(t) = \min_{k \in \mathcal{G}_g} \tilde{d}({}^0p_k(t); \mathcal{C}(t)), g = 1, \dots, G,$$

and the nearest human point ${}^0\hat{p}_g^{\text{hum}}$ is the projector $\pi({}^0p_{\hat{k}}; a_{\hat{c}}, b_{\hat{c}})$ at the attaining pair $(\hat{k}, \hat{c})$.

4.2.5 Smooth minimum

For differentiability and noise rejection, a soft minimum can replace the hard min:

$$\text{smin}_\tau\{x_i\} = -\tau \log \sum_i e^{-x_i/\tau}, \tau > 0$$

yielding $d_g^{\text{s}} = \text{smin}_\tau\{\tilde{d}({}^0p_k; \mathcal{C})\}_{k \in \mathcal{G}_g}$ and $d_{\min}^{\text{s}} = \text{smin}_\tau\{\tilde{d}({}^0p_k; \mathcal{C})\}_{k=1..N_r}$. As $\tau \to 0^+, \text{smin}_\tau \to \min$.

4.2.6 Nearest-pair witnesses

Along with the scalar distances, the query returns the witness pair for every group:

$$(\hat{p}_g^{\text{rob}}, \hat{p}_g^{\text{hum}}) = \arg\min_{p_k, c} \tilde{d}(p_k; (a_c, b_c, r_c))$$

used downstream to define repulsive directions $\hat{n}_g = \frac{{}^a p_g^{\text{rob.}} - {}^a p_g^{\text{hum}}}{\| {}^a p_g^{\text{rob}} - {}^a p_g^{\text{hum}} \|}$.

4.2.7 Computational budget

The projection $\pi(\cdot)$ is closed-form and $O(1)$. With N_r robot points and N_c human proxies, a full scan is $O(N_r N_c)$ per cycle. In practice: cache capsule endpoints per human frame, stream robot point positions once per controller tick, early-exit per

group after a guard distance is crossed, and employ the hard minimum for triggering while using the soft minimum only where gradients are needed [21, 18, 17].

4.2.8 Outputs

The query provides $d_{\min}(t)$ for safety gating, the vector $\{d_g(t)\}$ for group-structured shaping, and the witness pairs $\{\hat{p}_g^{\text{rob}}, \hat{p}_g^{\text{hum}}\}$ for constructing repulsive task references and for logging.

4.3 Repulsive safety fields (logistic and reciprocal shaping)

4.3.1 Problem setup

For each link group $g = 1, \dots, G$, let $\hat{n}_g(t) \in \mathbb{R}^3$ be the unit vector from the closest human witness to the closest robot witness (from 4.2), and let $d_g(t) \geq 0$ be the corresponding clearance. A repulsive Cartesian reference for group g is

$$^0v_g(t) = v\big(d_g(t)\big)\hat{n}_g(t)$$

where $v(\cdot)$ is a scalar speed law that is monotonically decreasing in d, bounded, and differentiable on $(0, +\infty)$.

4.3.2 Shaping laws

Two families are used depending on the desired falloff and saturation characteristics:

1. Logistic (sigmoidal) law

With parameters $V_{\max} > 0$ (speed cap), $r_{\text{if}} > 0$ (inflection-range proxy), and $\alpha > 0$ (steepness),

$$v_{\log}(d) = \frac{V_{\max}}{1 + \exp\left(\alpha\left(\frac{2d}{r_{\text{if}}} - 1\right)\right)}$$

Properties: $v_{\log}(0) \approx V_{\max}, v_{\log}(r_{\text{if}}/2) = \frac{V_{\max}}{2}, \lim_{d\to+\infty} v_{\log}(d) = 0, \frac{dv_{\text{cg}}}{dd}$ is bounded.

2. Reciprocal (inverse-distance) law with taper

With gain $k > 0$, taper distance $r_{\text{act}} > 0$, and small $\varepsilon > 0$,

$$v_{\text{rec}}(d) = \text{clip}_{[0,V_{\max}]}\left(k\left(\frac{1}{d+\varepsilon} - \frac{1}{r_{\text{act}}+\varepsilon}\right)\right)$$

which is positive for $d < r_{\text{act}}$ and zero otherwise, then saturated at $V_{\max}$. This yields a long tail near contact and a hard activation at r_{act} (Khatib, 1986; Merckaert et al., 2022).

4.3.3 Group weighting and span mapping

Let $\mathcal{G}_g$ denote the set of robot witnesses for group g (proximal → distal ordering), and let the "reference-style" span for group g be the first s_g joints (e.g., $s_g = g + 1$ on a 7 -DoF arm). Define a $3 \times s_g$ point Jacobian evaluated at the group's closest robot witness,

$$J_g^{(p)}(q) = \left[\omega_1 \times (r_g - o_1) \quad \cdots \quad \omega_{s_g} \times \left(r_g - o_{s_g}\right)\right],$$

where ω_j and o_j are the j-th joint axis and origin in the world frame, and r_g is the closest robot witness position for group g. The raw joint-rate contribution for group g is

$$\dot{q}_g^{\text{raw}}(q) = \kappa_g\left(J_g^{(p)}(q)\right)^{\top} v_g,$$

with $\kappa_g > 0$ a dimensionless weight (per-group gain).

4.3.4 Combination and null-space projection

Summing over all groups and projecting where appropriate gives

$$\dot{q}_{\text{rep}}^{\text{raw}} = \sum_{g=1}^{G} \dot{q}_g^{\text{raw}}, \qquad \dot{q}_{\text{rep}} = S_q\left(\dot{q}_{\text{rep}}^{\text{raw}}\right),$$

where $S_q(\cdot)$ is a joint-rate limiter enforcing $|\dot{q}_i| \le \dot{q}_i^{\max}$. In posture-biased modes, $\dot{q}_{\text{rep}}$ is superposed with the nominal posture bias; in fixed-TCP modes (Chapter 4.5), it is injected through the task null space,

$$\dot{q} = \dot{q}_{\text{task}} + N(q)\dot{q}_{\text{rep}}, \qquad N(q) = I - J^{\#}(q)J(q),$$

with J the $6 \times n$ geometric Jacobian of the TCP and $J^{\#}$ a damped pseudoinverse.

Speed and joint caps are enforced before inversion and projection to keep behavior consistent across modes (see §3.4.3).

4.3.5 Distance-to-velocity direction

Given the witness pair ($\hat{p}_g^{\text{rob}}, \hat{p}_g^{\text{hum}}$),

$$\hat{n}_g = \frac{\hat{p}_g^{\text{rob}} - \hat{p}_g^{\text{hum}}}{\left\|\hat{p}_g^{\text{rob}} - \hat{p}_g^{\text{hum}}\right\| + \epsilon_n},$$

with $\epsilon_n > 0$ for numerical robustness; $\hat{n}_g$ always points away from the human proxy.

4.3.6 Saturation and smoothness considerations

- **Speed capping:** $V_{\max}$ bounds the Cartesian magnitude per group.
- **Joint capping:** $S_q(\cdot)$ enforces joint-wise limits and prevents windup.
- **Differentiability:** the logistic law is C^{∞} for $d > 0$; the reciprocal law is C^{∞} on $(0, +\infty)$ and Lipschitz at r_{act} after clipping.
- **Multi-group coherence:** proximal groups typically use higher κ_g and smaller r_{act} (or r_{if}) to bias evasive motion toward upstream joints.

4.3.7 Parameters and defaults

$V_{\max}\left(\frac{\text{m}}{\text{s}}\right)$, $\alpha(-)$, $r_{\text{if}}(\text{m})$, $k(\text{m}^2/\text{s})$, $r_{\text{act}}(\text{m})$, $\{\kappa_g\}_{g=1..G}(-)$, joint-rate bounds $\{\dot{q}_i^{\max}\}(\text{rad/s})$, and numerical epsilons ε, ϵ_n.

4.3.8 Outputs

At each control tick: (a) group clearances d_{g_1} (b) repulsive TCP-space references ${}^{0}v_{g_2}$ (c) joint-space contribution $\dot{q}_{\text{rep}}$ ready for null-space injection or posture superposition, and (d) capped diagnostic quantities for logging (speed utilizations, active groups, and per-group saturations).

4.4 SSM-style supervisor: STOP/RELEASE hysteresis and dwell

4.4.1 Objective

Coordinate the safety behaviors of Section 4.3 with the nominal TCP task so that pausing, repelling, and resuming are deterministic, chatter-free, and compatible with time laws (vector and LSPB).

4.4.2 State set and outputs

Let the discrete state be

$$x \in \mathcal{X} = \{ \text{Approach, Hold, Repel, Resume, Stop} \}.$$

At each control tick, the commanded joint rate is

$$\dot{q}_{\text{cmd}} = \begin{cases} \dot{q}_{\text{task}} & x = \text{ Approach}, \\ 0 & x = \text{ Hold or Stop}, \\ \dot{q}_{\text{task}} + N(q)\dot{q}_{\text{rep}} & x = \text{ Repel}, \\ \Pi_{\text{resume}}(\dot{q}_{\text{task}}) & x = \text{ Resume}, \end{cases}$$

where $\dot{q}_{\text{task}}$ is the nominal (vector or LSPB) command, $N(q) = I - J^{\#}J$ the TCP null-space projector, $\dot{q}_{\text{rep}}$ the repulsive contribution from 4.3, and Π_{resume} the mode-dependent resume policy (below).

The mode-aware command path is organized as shown in Figure 4.1:

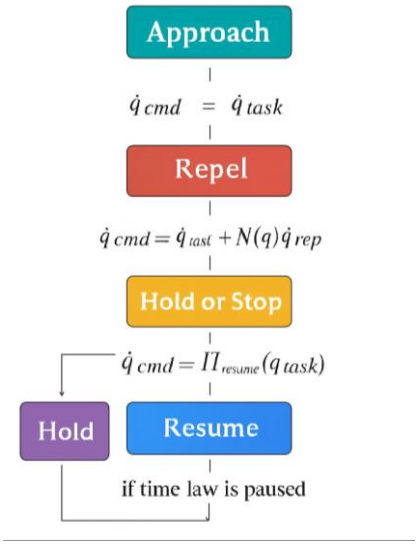

Fig. 4.1 Command computation by mode. Approach: q̇_cmd = q̇_task; Hold/Stop: q̇_cmd = 0; Repel: q̇_cmd = q̇_task + N(q) q̇_rep; Resume: q̇_cmd = Π_resume(q̇_task). Inputs: d_min, guards, timers; core blocks: distance query → supervisor → projection N(q) = I − J#J.

4.4.3 Clearance aggregates and thresholds

With per-group clearances $\{d_g\}_{g=1..G}$ and group weights $\{\kappa_g\}$, define the global minimum and a weighted surrogate:

$$d_{\min} = \min_g d_g, \qquad \tilde{d} = \min_g \left(d_g/\sqrt{\kappa_g}\right).$$

Two radii implement hysteresis:

$$r_{\text{stop}} < r_{\text{release}}$$

and two dwell times complete the guard set:

$$T_{\downarrow} > 0 \text{ (enter-stop dwell)}, \quad T_{\uparrow} > 0 \text{ (release dwell)}.$$

The chosen bands and dwell were tuned to minimize chattering; measured stop/resume statistics are reported in Chapter 6.

4.4.4 Guards and timers

Let $t_{\downarrow}$ and $t_{\uparrow}$ be timers (reset to 0 when their condition is not met). The guards are

$$\mathcal{G}_{\text{stop}} : d_{\min} \le r_{\text{stop}} \ \wedge\ t_{\downarrow} \ge T_{\downarrow}$$
$$\mathcal{G}_{\text{release}} : d_{\min} \ge r_{\text{release}} \ \wedge\ t_{\uparrow} \ge T_{\uparrow}$$

Timers evolve as

$$\dot{t}_{\downarrow} = \begin{cases} 1 & d_{\min} \le r_{\text{stop}} \\ 0 (t_{\downarrow} \leftarrow 0) & \text{otherwise} \end{cases}, \quad \dot{t}_{\uparrow} = \begin{cases} 1 & d_{\min} \ge r_{\text{release}} \\ 0 (t_{\uparrow} \leftarrow 0) & \text{otherwise.} \end{cases}, \quad updated\ s(t)$$

4.4.5 Transitions

The finite-state logic is:

- Approach → Repel if $d_{\min} < r_{\text{release}}$ and $\tilde{d} < r_{\text{release}}$ (activate repulsion before the stop band).
- Repel → Stop if $\mathcal{G}_{\text{stop}}$ is true.
- Stop → Resume if $\mathcal{G}_{\text{release}}$ is true.
- Resume → Approach after the resume policy completes (below) and $d_{\min} \ge r_{\text{release}}$ holds during the policy.
- Hold is a transient freeze used by the LSPB pause semantics: Approach/Repel → Hold when the time law is paused; Hold → Resume when resuming that law.

The finite-state logic in this work is summarized by the supervisory state machine below:

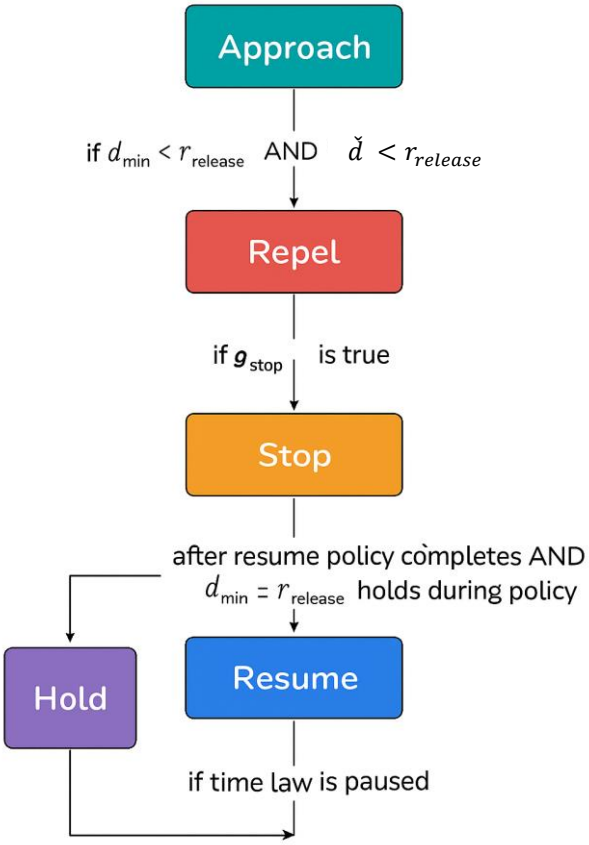

Fig. 4.2 Finite-state supervisor with hysteresis radii (r_stop, r_release) and dwell timers (T↓, T↑). Transitions are guarded by G_stop and G_release; Hold freezes the time law, Resume re-enables it after the policy completes.

4.4.6 Pause/resume semantics by time law

Vector approach (scenarios 1 & 2): no global clock; Resume simply re-enables $\dot{q}_{\text{task}}$ immediately,

$$\Pi^{\text{vec}}_{\text{resume}}(\dot{q}_{\text{task}}) - \dot{q}_{\text{task}}$$

LSPB time law (scenarios 3 & 4): the phase variable $s \in [0,1]$ is frozen in Stop/Hold, i.e., $\dot{s} - 0$. On Resume, the LSPB restarts from the last phase $\bar{s}$ with bounded jerk:

$$s(t) - \bar{s} + \int_{t_0}^{t} \dot{s}(\tau)d\tau, \qquad \dot{s} - \text{lspb}(\bar{s} \to 1; a_{\max}, v_{\max})$$

and TCP velocity ramps with a C^1 splice so that the commanded $\dot{x}$ remains bounded. During Resume, Π_{resume} returns the LSPB-derived $\dot{q}_{\text{task}}$ consistent with the updated $s(t)$.

For vector references, "Resume" simply re-enables the nominal twist; for LSPB, the phase s continues from its frozen value with jerk-bounded splice.

4.4.7 Arbitration with repulsion

Repulsion remains active in Repel and is suppressed in Stop/Hold. In Resume, repulsion is allowed but limited so as not to corrupt the primary task; specifically,

$$\left\| JN(q)\dot{q}_{\text{rep}} \right\| \leq \lambda_{\text{leak}} \left\| \dot{x}_{\text{task}} \right\|, \qquad 0 < \lambda_{\text{leak}} \ll 1,$$

which bounds task-space leakage from null-space action.

4.4.8 Chatter avoidance and guarantees

The strict inequality $r_{\text{stop}} < r_{\text{release}}$ plus $T_{\downarrow}, T_{\uparrow} > 0$ yields a two-sided hysteresis with temporal deadbands: repeated Stop-Resume oscillations are excluded for bounded clearance rates. Under bounded sensing/actuation latencies, Stop is triggered no later than $T_{\downarrow}$ after d_{min} first crosses r_{stop}, and Resume occurs no earlier than $T_{\uparrow}$ after d_{min} re-enters the safe band.

4.4.9 Logged indicators for evaluation

At each tick the supervisor logs x, d_{min}, active group index, timers $t_{\downarrow}, t_{\uparrow}$, and LSPB phase s (when applicable). These feed the dwell-time statistics, pause durations, and restart smoothness metrics reported in chapter 6.

4.5 Fixed-TCP avoidance (6×7) and orientation locking

4.5.1 Objective

Exploit kinematic redundancy to keep the TCP pose intact while reshaping the arm posture away from the human. When small orientation drift is acceptable, apply a soft clamp that holds the TCP attitude within a narrow deadband.

4.5.2 Task definition

Let the TCP twist be $\dot{x} = [v^\top \quad \omega^\top]^\top \in \mathbb{R}^6$, the geometric Jacobian $J(q) \in \mathbb{R}^{6\times 7}$, and the joint velocity $\dot{q} \in \mathbb{R}^7$. The nominal TCP regulation uses a damped least-squares pseudoinverse

$$J^{\#}(q) = W^{-1}J^\top(JW^{-1}J^\top + \lambda^2 I_6)^{-1}$$

with positive-definite joint weighting $W \succ 0$ and damping $\lambda \geq 0$ (possibly scheduled with the manipulator's conditioning). The null-space projector is

$$N(q) = I_7 - J^{\#}(q)J(q)$$

4.5.3 Fixed-TCP avoidance

In fixed-TCP avoidance the commanded TCP twist is zero,

$$\dot{x}_{\text{task}} = 0, e_x = 0$$

so the only admissible motion lies in null(J). Let $\dot{q}_{\text{rep}}$ be the 7×1 repulsive joint-rate proposal produced by Section 4.3 (before any projection). The fixed-TCP command is

$$\dot{q}_{\text{cmd}} = N(q)\dot{q}_{\text{rep}}$$

By construction, $J\dot{q}_{\text{cmd}} = 0$, so the TCP is kinematically invariant.

4.5.4 Leak clipping

Finite precision, model mismatch, and latency can introduce residual task-space motion $r = J\dot{q}_{\text{cmd}}$. Enforce a strict bound $\|r\| \leq \varepsilon_{\text{leak}}$ by scaling:

$$\beta = \min\left(1, \frac{\varepsilon_{\text{leak}}}{\|JN(q)\dot{q}_{\text{rep}}\| + \delta}\right), \qquad \dot{q}_{\text{cmd}} \leftarrow \beta N(q)\dot{q}_{\text{rep}}$$

with small $\delta > 0$ for numerical safety. Optionally, recompute r after scaling and zero any residual using a short corrective step $\Delta\dot{q} = -J^{\#}r$; in fixed-TCP mode this reduces to a second-order effect and is typically not needed if $\varepsilon_{\text{leak}}$ is tight. The corresponding LEAK_EVT and scale factor are logged each tick (see §3.5.7).

4.5.5 Orientation locking (soft clamp)

When the position must be held and the orientation should remain within a small tube about a reference $R_{\text{ref}} \in \text{SO}(3)$, we use a gentle orientation error feedback, which activates only outside a deadband. With current orientation R, define the skew error

$$E_R = \frac{1}{2}\left(R_{\text{ref}}^{\top} R - R^{\top} R_{\text{ref}}\right), \qquad e_\omega = \text{vee}(E_R) \in \mathbb{R}^3.$$

Let $\omega_{\max} > 0$ and a deadband $\theta_0 > 0$. The orientation clamp twist is

$$\omega_{\text{clamp}} = -k_\omega \text{sat}_{\omega_{\max}}\left(\psi(e_\omega; \theta_0)\right), \qquad v_{\text{clamp}} = 0$$

where $\psi(\cdot; \theta_0)$ smoothly gates the error to zero for $\|e_\omega\| \le \theta_0$ (e.g., a cubic deadzone), and sat $_{\omega_{\max}}$ limits magnitude. The combined twist command in "orientation-locked" fixed-TCP mode is

$$\dot{x}_{\text{task}} = \begin{bmatrix} 0 \\ \omega_{\text{clamp}} \end{bmatrix}, \qquad \dot{q}_{\text{task}} = J^{\#} \dot{x}_{\text{task}}$$

Repulsion remains null-space-only:

$$\dot{q}_{\text{cmd}} = \dot{q}_{\text{task}} + N(q)\dot{q}_{\text{rep}},$$

with the same leak clipping on $JN\dot{q}_{\text{rep}}$ and, if desired, a fractional cap

$$\left\| JN(q)\dot{q}_{\text{rep}} \right\| \le \lambda_{\text{leak}} \left\| \dot{x}_{\text{task}} \right\|, \qquad 0 < \lambda_{\text{leak}} \ll 1,$$

to ensure the clamp remains dominant whenever it is active. When the clamp is active, null-space repulsion is limited by $\|JN\dot{q}_{rep}\| \le \lambda_{leak}\|\dot{x}_{task}\|$ with $0 < \lambda_{leak} \ll 1$.

4.5.6 Null-space shaping and limits

Repulsion can be augmented with standard posture shaping in the null space without affecting TCP invariance. For a joint-limit barrier potential $U(q) = \sum_i u_i(q_i)$ with gradient ∇U, include

$$\dot{q}_{\text{ns}} = -K_U \nabla U(q), \dot{q}_{\text{cmd}} \leftarrow \dot{q}_{\text{cmd}} + N(q)\dot{q}_{\text{ns}}$$

and finally apply joint-space rate/acceleration saturations before execution.

4.5.7 Computational notes

All projections use the current J and $J^{\#}$ evaluated at the measured q. Damping λ and weights W should mirror those in Chapter 3 to preserve numerical conditioning; a typical choice is $W = \mathrm{diag}(w_i)$ with higher weights on distal joints to favor proximal reconfiguration.

Weights and damping mirror Chapter 3 to keep the projector numerically aligned across scenarios. The deadband θ_0, gains k_ω, and leak bounds $\varepsilon_{\text{leak}}$ are declared in Section 3.5 and reused here for consistency.

4.6 Stability and transparency considerations

4.6.1 Objectives

- preserving the primary task; TCP motion must follow the commanded twist (or remain fixed in 4.5) despite avoidance;
- bound the closed-loop inputs so joint limits, rates, and accelerations are respected;
- keep the interaction predictable to an operator observing the TCP (transparency).

Transparency is evaluated from logs via pause duration, restart smoothness, and leakage events (see Chapter 6).

4.6.2 Task preservation under null-space shaping

With the velocity command

$$\dot{q} = J^{\#}(q)\dot{x}_{\text{task}} + N(q)\dot{q}_{\text{ns}}, \qquad N(q) = I_7 - J^{\#}J$$

the induced TCP twist is

$$\dot{x} = J\dot{q} = JJ^{\#}\dot{x}_{\text{task}} + JN\dot{q}_{\text{ns}} = \Pi_J \dot{x}_{\text{task}}$$

where $\Pi_J = JJ^{\#}$ is an idempotent projector onto range (J). If $\dot{x}_{\text{task}} \in \text{range}(J)$ (nominal case), then $\Pi_J \dot{x}_{\text{task}} = \dot{x}_{\text{task}}$ and $JN = 0$; hence null-space terms do not corrupt the task. In the fixed-TCP mode of 4.5, $\dot{x}_{\text{task}} = 0$ and $\dot{x} = 0$ by construction.

4.6.3 DLS conditioning and bounded joint rates

The damped pseudoinverse

$$J^{\#} = W^{-1}J^{\top}(JW^{-1}J^{\top} + \lambda^2 I_6)^{-1}$$

regularizes near singularities and yields the bound

$$\|J^{\#}\| \leq \frac{1}{\lambda}\left\|W^{-1/2}\right\|$$

so for any bounded $\dot{x}_{\text{task}}$ we obtain bounded $\dot{q}$. Scheduling $\lambda = \lambda(\sigma)$ as a nondecreasing function of a conditioning index σ (e.g., manipulability) prevents rate blow-up while limiting task distortion.

4.6.4 Repulsion boundedness and saturation

Repulsive references are generated as bounded linear velocities in world frame, then mapped to joints by $J^{\top}$ or point Jacobians. Denote a per-link bound $\left\|V_{\text{rep}}\right\| \leq V_{\max}$; with Jacobian columns J_p and joint-rate cap $\dot{q}_{\max}$,

$$\left\|\dot{q}_{\text{rep}}\right\| \leq \left\|J_p^{\top}\right\| V_{\max} \;\Rightarrow\; \dot{q}_{\text{ns}} = \text{sat}_{\dot{q}_{\max}}\left(\dot{q}_{\text{rep}}\right)$$

Axis-wise rate and acceleration limiters enforce bounded joint inputs regardless of distance-field peaks (caps applied pre-inversion; see §3.4.3).

4.6.5 Leakage control and small-gain rationale

Null-space components can leak into the task through discretization, latency, and Jacobian mismatch. Let $r = JN\dot{q}_{\text{ns}}$. The command applies a scaling $\beta \in (0,1]$ such that $\|r\| \leq \varepsilon_{\text{leak}}$. The closed-loop task channel becomes

$$\dot{x} = \Pi_J \dot{x}_{\text{task}} + r, \qquad \|r\| \leq \varepsilon_{\text{leak}}$$

Choosing $\varepsilon_{\text{leak}}$ below the measurement/quantization floor renders repulsion effects second order in the task dynamics (small-gain argument). In orientation-locked

mode, an additional fractional cap $\|r\| \leq \lambda_{\text{leak}} \|\dot{x}_{\text{task}}\|$ preserves clamp dominance when active.

Discrete-time implementation and passivity hints at sampling time T_s, the joint update is $\Delta q = T_s \dot{q}_{\text{cmd}}$. Stability requires consistent timing and filtered references. Two practical measures:

- **first-order hold for repulsion:** $V_{\text{rep}}[k] = \alpha V_{\text{rep}}[k-1] + (1-\alpha)\hat{V}[k]$ with $\alpha = e^{-T_s/\tau}$, suppressing high-frequency injections;
- **energy consistency:** cap the incremental joint power $P_k = \tau_k^{\top} \dot{q}_k$ using a tank-like budget or simply limit $\|\dot{q}_k\|$ adaptively when large external corrections (e.g., STOP/RELEASE transitions) occur. These steps mitigate discrete-time active behavior near steep distance gradients.

4.6.6 Hysteresis and dwell for mode transitions

Binary supervisors (STOP/RELEASE) and soft states (Approach/Hold/Repel/Resume) employ distance hysteresis ($d_{\text{release}} > d_{\text{stop}}$) and dwell timers. This eliminates chatter, avoids rapid sign flips in V_{rep}, and ensures that the effective joint command remains piecewise-smooth. With bounded $\dot{q}$ and minimum dwell $t_{\min}$, the number of switches on any finite interval is finite, guaranteeing well-posed execution.

4.6.7 Transparency to the operator

Transparency is maintained when the TCP trajectory is either preserved (fixed-TCP) or altered only within explicit, bounded envelopes. The design enforces:

- invariance or near-invariance of the commanded TCP path (via projection and leakage caps);
- bounded, smooth posture motions (via filtering and saturations);
- predictable supervisory behavior (via hysteresis and dwell).

Together, these yield operator-observable behavior that is consistent with the nominal task while ensuring separation from the human skeleton proxies.

4.7 Conclusions

Chapter 4 has established the human–robot interface that underpins the safety logic used throughout the thesis. We specified a skeleton-to-capsule lifting pipeline with health flags and explicit frame semantics; defined nearest-pair distance queries and the derived safety variables (clearances, hysteresis thresholds, dwell timers); and introduced two complementary behaviors: a continuous repulsive field blended with posture shaping, and an SSM-style supervisor whose STOP/RELEASE actions include dwell to pause and resume an LSPB time law without corrupting its schedule. Critically, all corrective actions are constrained to the Jacobian null space, with leak bounds and joint-limit/manipulability safeguards, so that task-space intent—including the fixed-TCP option—remains preserved while proximity risk is mitigated.

Beyond detailing mechanisms, the chapter made the contracts explicit: what the motion layer expects from the distance layer (rates, units, validity), what the supervisor guarantees to the reference generator (monotonic timing with dwell), and what logs must be emitted for auditability. The result is a small, typed interface of safety variables that is implementation-ready and testable, making failure modes observable (timeouts, range violations) and recovery predictable. We also clarified the limits of the approach—e.g., sensitivity to skeleton quality and conservative clearances—and pointed to mitigations (health gating, hysteresis, dwell semantics, posture bias) that stabilize behavior near decision boundaries.

Chapter 5

Implementation & Software Architecture (CoppeliaSim)

This chapter documents the implementation and software architecture used to realize the methods in simulation. The work is conducted in CoppeliaSim, a physics-based robotics environment that provides deterministic synchronous stepping, an extensible scene graph for articulations and sensors, and a remote interface for coupling external controllers. It is selected here because it allows the robot, human proxy, and safety supervisor to run under a single simulation clock while exposing low-level kinematic and geometric data needed for online Jacobians, distance queries, and visualization.

The scene models a 7-DoF manipulator mounted on a work surface, a parameterized human proxy built from jointed segments, and a target frame serving as the task reference. Along the manipulator, a set of lightweight "control spheres" is attached to selected links to act as geometric samples for distance computations. Each articulated element publishes its pose with respect to the world frame, so that the controller can reconstruct joint screw axes and point Jacobians without peeking into the simulator's internal solvers. The human proxy's joints are driven either from motion-capture frames or scripted motions, and are aligned to the robot's world frame via a fixed transform consistent with the data pipeline used in Chapter 4.

The controller runs in MATLAB and communicates with CoppeliaSim through the remote API over a TCP/IP session. All streams are world-aligned and time-stamped at the control tick so reconstruction remains deterministic across runs (see §5.4).

The simulator operates in synchronous mode: every simulation step triggers a sensing–control–actuation handshake. Tick indices, not wall-clock time, are used as the primary key for logs and latency histograms (see §5.4.1)

On each tick, the simulator emits three compact data streams: (i) the world positions of the control spheres, (ii) the world orientations of the robot links, and (iii) full per-link poses, including the gripper, for visualization and alignment checks. MATLAB subscribes to these streams, reconstructs the geometric Jacobian from the streamed

frames, evaluates the task command under the active trajectory time law, computes avoidance actions in task or null space as appropriate, and returns joint commands that are applied on the next simulation step [1, 2]. This closed loop ensures that physics integration, measurements, and control share one clock and that latency is both bounded and measurable.

Within this framework, the simulator is the authoritative source of ground-truth kinematics and geometry, while the external controller remains model-based but measurement-driven. The time law for the end-effector is interchangeable: a continuous vector-field reference can be layered directly on measured frames, or a trapezoidal (linear–segment with parabolic blends) time law can be used to enforce acceleration and velocity limits with pause–resume semantics for the safety supervisor. Safety behaviors are realized in two complementary ways. When the end-effector task must be preserved, avoidance is projected into the manipulator's null space so that the primary task remains uncorrupted. When timing guarantees are paramount, a supervisory finite-state logic pauses and resumes the time law according to stop/release thresholds and dwell times. Both behaviors consume the same distance queries against human capsules, evaluated in the world frame with explicit units.

All behavioral modes share a single software backbone: nominal tracking without human interaction, continuous repulsion layered on tracking, pause–resume supervision around timing laws, and fixed-TCP operation with null-space avoidance. Every run is captured end to end. Inputs, outputs, and state variables are timestamped; logging includes units and coordinate frames; configuration files, random seeds, and code/version hashes are stored with the data; and scene assets and rates are summarized in manifests. The result is a controlled, time-deterministic environment in which human modeling, distance queries, safety behaviors, and trajectory time laws can be exercised, compared, and reproduced without ambiguity.

5.1 Scene and synchronization

This work uses CoppeliaSim in remote synchronous mode, with MATLAB as the external controller that advances the simulator exactly one physics step per control tick. The scene contains: (i) the robot with auxiliary point markers attached along its links for proximity and Jacobian-point evaluation; (ii) a kinematic human mannequin actuated at shoulders, elbows, spine, abdomen, and head; and (iii) a target frame for the tool center point (TCP). Unless otherwise stated, all poses are expressed in the world frame W.

The simulation scene used throughout this thesis comprises the 7-DoF manipulator with link-mounted control spheres, a kinematic human mannequin actuated at shoulder, elbow, spine, abdomen, and head, and a world-fixed TCP target frame.

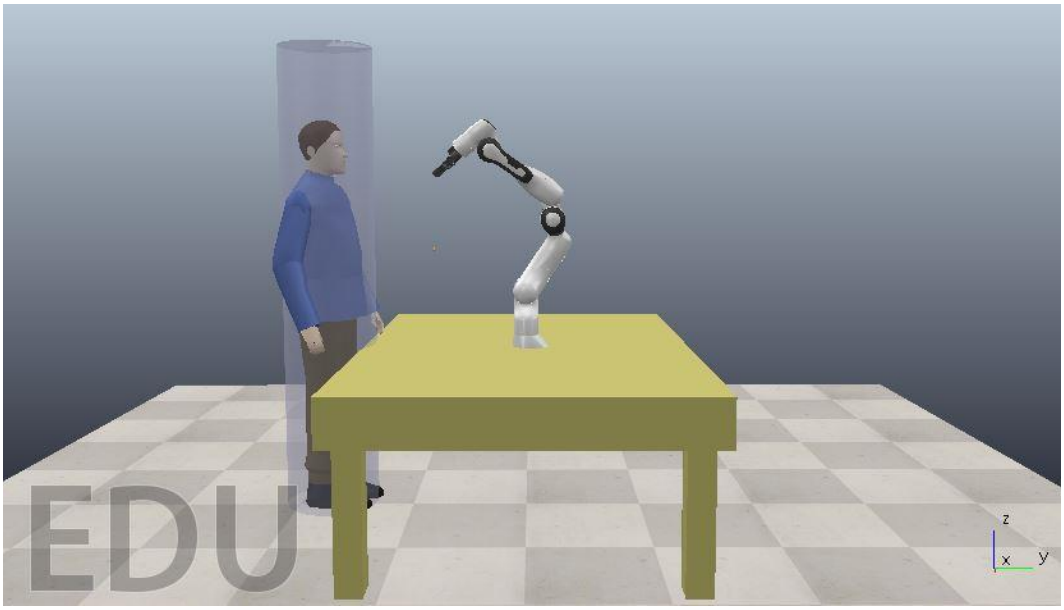

Fig. 5.1 CoppeliaSim scene: manipulator with control spheres, human mannequin, and TCP target; world frame W is the common reference for geometry and distance queries.

5.1.1 Frames and kinematic references

The robot base frame B is fixed to link 0. Link frames $\{L_i\}_{i=1}^{7}$ follow the manufacturer's convention; the TCP frame is T. For the human mannequin, anatomical joint frames are defined at shoulder, elbow, wrist, abdomen, spine, and head. The motion-capture skeleton is first rigidly aligned to the simulator world before joint-angle extraction. The alignment is a fixed homogeneous transform

$$A = \begin{bmatrix} R_z(\gamma) & p_0 \\ \mathbf{0}^\top & 1 \end{bmatrix}, \qquad \gamma = -135°, \qquad p_0 = \begin{bmatrix} -0.20 \\ -0.40 \\ 0.705 \end{bmatrix} \text{m}$$

If $p^{\text{mocap}} \in \mathbb{R}^3$ is a raw skeleton point, the aligned point is

$$\tilde{p}^W = R_z(\gamma) p^{\text{mocap}} + p_0$$

Joint angles for the mannequin are then obtained from $\{\tilde{p}^W\}$ via the geometric constructions described in Chapter 4 and streamed to CoppeliaSim.

5.1.2 Data exchange

Each simulation step publishes three streams required by the controller:

- world positions of the robot's control spheres $\{s_k\}$, concatenated as

$$S^W = [p(s_1)^\top \quad p(s_2)^\top \quad \dots \quad p(s_M)^\top]^\top \in \mathbb{R}^{3M}$$

- world orientations of the seven link frames, encoded as ZYX Euler triplets

$$E^W = [e(L_1)^\top \quad e(L_2)^\top \quad \dots \quad e(L_7)^\top]^\top \in \mathbb{R}^{21}$$

- compact link poses for visualization,

$$X^W = [x(L_1)^\top \quad \dots \quad x(L_8)^\top \quad x(\text{ gripper })^\top]^\top, x(L_i) = [p(L_i)^\top, e(L_i)^\top]$$

Conversely, the controller writes mannequin joint commands and the robot command (joint-space or task-space, depending on mode), and reads the TCP target position $p_T^\star$.

Table 5.1 summarizes the world-aligned data streams and command channels used throughout the experiments.

Source → Sink	Signal (symbol)	Dim	Units	Role / contents	Nominal rate
Simulator → Controller	S_W = [p(s1)^T ... p(sM)^T]^T	3M	m	World positions of control spheres {s_k} in W	1/Δt
Simulator → Controller	E_W = [e(L1)^T ... e(L7)^T]^T	21	rad (ZYX)	World orientations (Euler) of link frames {L_i}	1/Δt
Simulator → Controller	X_W = [x(L1)^T ... x(gripper)^T]^T	6×9	m, rad	Compact poses for visualization/sanity checks	1/Δt
Controller → Simulator	Robot command	7 (joints) or 6 (twist)	rad/s, m/s	Joint-space or task-space command (mode-dependent)	1/Δt

Controller → Simulator	Mannequin joints	scene-dependent	rad	Actuation of mannequin shoulder, elbow, spine, abdomen, head	1/Δt
Simulator → Controller	p_T*	3	m	Target position for TCP (when used)	1/Δt

Table 5.1 World-aligned data streams and commands between simulator and controller.

5.1.3 Synchronous stepping

The simulator runs in synchronous mode and advances only when triggered by the controller. Let Δt_{ctrl} denote the controller period and Δt_{phys} the physics integrator step. In this configuration,

$$\Delta t \triangleq \Delta t_{\mathrm{ctrl}} = \Delta t_{\mathrm{phys}},$$

and all discrete-time modules (trajectory generators, supervisors, filters) are designed with sampling time Δt . The loop at tick k proceeds as:

$$\text{read } \{\mathrm{S}_{\mathrm{k}}^{\mathrm{W}}, \mathrm{E}_{\mathrm{k}}^{\mathrm{W}}, \mathrm{X}_{\mathrm{k}}^{\mathrm{W}}, \mathrm{p}_{\mathrm{T,k}}^{\star}\},$$

$$\text{update mannequin commands from } \{\tilde{\mathrm{p}}_{\mathrm{k}}^{\mathrm{W}}\},$$
$$\text{evaluate robot control law for the active mode,}$$
$$\text{write robot and mannequin commands,}$$
$$\text{trigger one physics step.}$$

This establishes a one-to-one mapping between control ticks and physics steps, removing sampling jitter and nondeterminism.

5.1.4 Timing guarantees and overruns

Trajectory-time parameters (e.g., LSPB segment durations) are chosen as integer multiples of Δt; timers in the STOP/RELEASE supervisor count ticks, ensuring exact dwell times. If computation at tick k exceeds a prescribed budget, the controller applies a hold-last-safe policy at tick $k+1$: the previously issued robot command u_{k-1} is retained while a timing flag is logged. This yields:

$$u_k = \begin{cases} \hat{u}_k & \text{if } t_{\mathrm{comp},k} \le t_{\max} \\ u_{k-1} & \text{otherwise} \end{cases}$$

where $\hat{u}_k$ is the freshly computed command and $t_{\text{comp},k}$ the measured compute time. Timing is enforced at tick granularity: if the compute budget is exceeded, the controller applies a hold-last-safe policy and logs the overrun; STOP/RELEASE dwell timers are tick-indexed to guarantee exact semantics, see figure below.

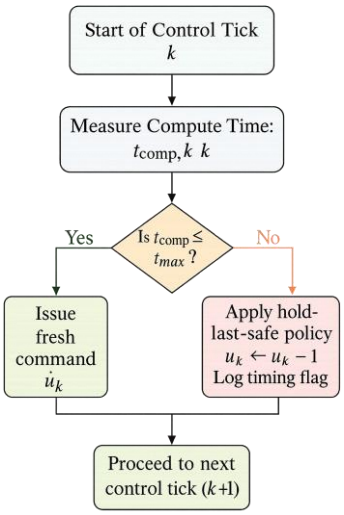

Fig. 5.2 Overrun policy and dwell accounting: when $t_{comp} \leq t_{max}$ the new command u_k is applied otherwise; the controller holds u_{k-1} and records the overrun; dwell timers advance per tick.

5.1.5 Validation hooks

At each tick the controller computes a pose-alignment residual for visualization sanity checks. If T_i^{exp} is the unpacked transform of link i and T_i^{vis} the local visualization transforms, the translational and rotational residuals are

$$r_i^{\text{pos}} = \left\| p\left(T_i^{\text{exp}}\right) - p\left(T_i^{\text{vis}}\right) \right\|_2, \qquad r_i^{\text{rot}} = \angle\left(R\left(T_i^{\text{vis}}\right) R\left(T_i^{\text{exp}}\right)^{\top}\right)$$

and are logged together with TCP pose, manipulability, minimum human-robot distance, and safety-state transitions. Overrun events are summarized as rates per minute and per thousand ticks in Chapter 6. Because these diagnostics are tied to the synchronous tick k, latency histograms and reproducibility reports later in the thesis are grounded in deterministic step indices.

Two primary checks are used each run: (i) FK↔IK round-trip pose residuals at the TCP and (ii) Jacobian consistency from streamed link frames.

5.2 Dataflow and helper primitives

The software stack is organized as a deterministic pipeline that maps sensed geometry into safe joint commands at each synchronous tick. The pipeline comprises six stages: (i) scene I/O, (ii) geometric lifting, (iii) distance queries, (iv) safety-field shaping and supervisory logic, (v) task–space tracking, and (vi) joint–space synthesis and post-processing. Each stage exposes a minimal, testable primitive; together they implement the five operating modes enumerated later in this chapter.

The six stages correspond to §5.2.1–§5.2.6; post-processing and test hooks are detailed in §5.2.7–§5.2.9.

Figure 5.3 summarizes the deterministic six-stage pipeline executed at each synchronous tick.

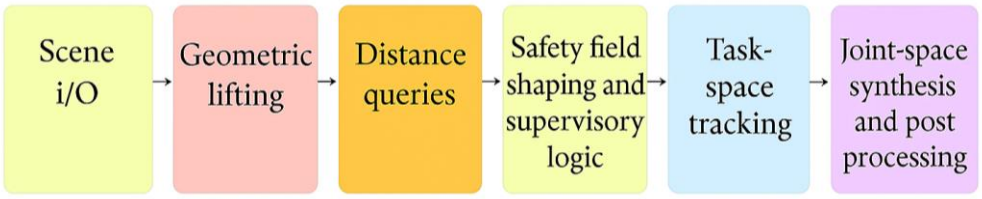

Fig. 5.3 Deterministic per-tick pipeline.

5.2.1 Scene I/O (world-aligned signals)

At tick k, the controller ingests:

$$\left\{S_k^W \in \mathbb{R}^{3M}, E_k^W \in \mathbb{R}^{21}, X_k^W, p_{T,k}^* \in \mathbb{R}^3, \tilde{P}_k^W \in \mathbb{R}^{3\times N_k}\right\},$$

namely the robot's control-point positions, link orientations, compact link poses for visualization, the current TCP target, and the world-aligned human skeleton points. The mannequin's joint targets (shoulders, elbows, spine, abdomen) are emitted to the simulator; the robot command is produced after the subsequent stages.

5.2.2 Geometric lifting (frames, Jacobians, kinematics)

From (E_k^W, X_k^W) the controller reconstructs the instantaneous kinematic map

$$f: \mathbb{R}^7 \to SE(3), T(q) = \begin{bmatrix} R(q) & p(q) \\ \mathbf{0}^\top & 1 \end{bmatrix}$$

and the geometric Jacobian $J(q) \in \mathbb{R}^{6\times 7}$. The world-linear velocity of any world point $r \in \mathbb{R}^3$ rigidly attached to link i is evaluated via point Jacobians:

$$v(r) = J_p(r,q)\dot{q}, J_p(r,q) = [\omega_1 \times (r - o_1) \quad \cdots \quad \omega_7 \times (r - o_7)],$$

where ω_j and o_j are, respectively, the world joint-axis direction and world joint origin for joint j. For the TCP, the 6D twist map is

$$\dot{x}_T = \begin{bmatrix} v_T \\ \omega_T \end{bmatrix} = J(q)\dot{q}$$

5.2.3 Distance queries (robot proxy points vs human capsules)

Proxy definitions and limb capsules follow Chapter 4; only minimum distances and their rates are consumed here.

The human model is represented by capsules $\mathcal{C}_\ell = \text{seg}(a_\ell, b_\ell) \oplus \mathbb{B}(0, r_\ell)$. Robot proximity is evaluated at the control points $\{s_m\}_{m=1}^M$. For each pair $(s_m, \mathcal{C}_\ell)$ the closest point on the segment and the raw Euclidean distance are

$$t^\star = \text{clip}_{[0,1]}\left(\frac{(s_m - a_\ell)^\top (b_\ell - a_\ell)}{\|b_\ell - a_\ell\|_2^2}\right), \qquad c_\ell = a_\ell + t^\star(b_\ell - a_\ell),$$
$$d_{\text{raw}}(m,\ell) = \|s_m - c_\ell\|_2, \qquad d_{\text{eff}}(m,\ell) = \max\{d_{\text{raw}}(m,\ell) - r_R - r_\ell, 0\}.$$

Per tick, the pipeline extracts both the global minimum $d_{\min} = \min_{m,\ell} d_{\text{raw}}(m,\ell)$ (for STOP/RELEASE logic) and, for each robot control group g, the best opposing pair $\left(s_{m_g}, c_{\ell_g}\right)$ to parameterize a local repulsive direction.

5.2.4 Safety-field shaping (repulsion in world and supervisor state)

A smooth, distance-to-speed shaping enforces bounded, continuous repulsion in world space:

$$v_{\text{rep}}(d) = \frac{V_{\max}}{1 + \exp\left(\alpha\left(\frac{2d}{\rho} - 1\right)\right)}, \alpha > 0, \rho > 0, V_{\max} > 0$$

Given the direction $n_g = \frac{\delta_{m_g} - c_{l_g}}{\left\| s_{m_g} - c_{l_g} \right\|_2}$ (zeroed if the norm is below a tolerance), the world repulsive linear velocity request for group g is

$$V_g^W = k_g v_{\text{rep}}\left(d_g^{\text{eff}}\right) n_g, \qquad k_g > 0$$

The supervisor maintains a finite state with hysteresis and dwell, driven by the raw global minimum $d_{\min}$:

Stop if $d_{\min} \le d_{\text{stop}}$, Release if $d_{\min} \ge d_{\text{rel}}$, $d_{\text{rel}} > d_{\text{stop}}$, with tick-accurate dwell timers to avoid chatter.

Speed and joint caps are enforced before IK inversion and null-space projection to keep behavior consistent across modes (§3.4.3).

5.2.5 Task-space tracking (vector and LSPB time laws)

Two time laws are supported for the TCP: (i) a vector field reference that directly specifies $\dot{x}_T^\star = [v_T^\star; \omega_T^\star]$ toward a moving attractor with smooth speed schedules; and (ii) a piecewise-linear with parabolic blends (LSPB) time law that parameterizes the scalar progress $s \in [0,1]$ along a path $x_T(s)$ with

$$\dot{s}(t) = \begin{cases} at, & 0 \le t < t_a, \\ \dot{s}_c, & t_a \le t \le t_b, \\ -a\left(t_f - t\right), & t_b < t \le t_f, \end{cases} \quad 0 \le \dot{s}(t) \le \dot{s}_{\max}, \qquad |\ddot{s}(t)| \le a_{\max},$$

leading to $\dot{x}_T^* = J_x(s)\dot{s}$ where $J_x(s) = \partial x_T / \partial s$. When the supervisor is in Stop, $\dot{s} = 0$ (pause semantics). Upon Release, the clock resumes without re-timing, preserving LSPB timing integrity.

For vector-field references, RESUME re-enables the nominal twist computed from the current attractor and speed law.

5.2.6 Joint-space synthesis (primary task + redundancy behaviors)

The commanded joint rates result from a strict composition rule. The primary 6D task (TCP tracking) uses damped least squares:

$$\dot{q}_{\text{pri}} = J(q)_\lambda^{\#} \dot{x}_T^\star, \qquad J_\lambda^{\#} = J^\top (JJ^\top + \lambda^2 I_6)^{-1},$$

with λ scheduled against manipulability or conditioning. Redundant behaviors (repulsion, posture bias) are confined to the null space:

$$N = I_7 - J_\lambda^{\#} J, \qquad \dot{q}_{\text{ns}} = \sum_g J_p\left(s_{m_g}, q\right)^{\top} V_g^W + K_{\text{post}}\,(q - q_{\text{nom}}).$$

The composite command before limits is

$$\dot{q}_{\text{raw}} = \dot{q}_{\text{pri}} + N\dot{q}_{\text{ns}}$$

When the supervisor enters Stop, $\dot{q}_{\text{raw}} \leftarrow \mathbf{0}$ (hold), while in Release the same law resumes with the current $J(q)$ and $\dot{x}_T^{\star}$.

Leakage $\|J(q)\,N(q)\,\dot{q}_{ns}\|$ is monitored and clamped under LEAK_THR; flags are logged each tick (§3.5.7).

5.2.7 Post-processing (limits, smoothing, discretization)

The raw joint rates are passed through saturation and discrete-time smoothing consistent with the sampling time Δt :

$$\dot{q}_k = \text{clip}\big(\dot{q}_{\text{raw},k}, -\dot{q}_{\max}, \dot{q}_{\max}\big), \qquad q_{k+1} = q_k + \Delta t\,\dot{q}_k,$$

optionally with a first-order rate filter to cap $\ddot{q}$. All saturations and state transitions emit health flags and timestamps for later analysis.

5.2.8 Mannequin joint extraction (skeleton to joint commands)

For each tick, the aligned skeleton $\tilde{P}_k^W$ yields orthonormal frames at shoulders, elbows, spine, and abdomen by geometric constructions (differences, cross products, normalization). These frames are converted to the simulator's joint parameterization (e.g., spherical-joint direction-cosine matrices for shoulders; single-axis angles for elbows). Let R_{seg}^W denote a segment frame; the transmitted parameter vector is a compact embedding of $R_{\text{seg}}^{\text{W}}$ required by the mannequin joints. The same alignment transform guarantees spatial consistency between human proxies and robot/world coordinates.

5.2.9 Determinism and test hooks

Each primitive exposes tick-indexed inputs/outputs and admits unit checks: Jacobian symmetry tests, finite-difference vs analytic derivatives, capsule distance regression against synthetic cases, boundedness of $v_{rep}(d)$, and null-space leakage monitors $||J\,\dot{q}_{ns}||_2$. Because all stages run within a single synchronous step, the recorded traces map unambiguously to controller ticks, enabling reproducible experiments and latency budgeting reported later.

Any randomized elements use a fixed RNG seed recorded in the run configuration and artifacts (§5.4.1).

5.3 Mode scripts: behavior mapping

This section enumerates the operating modes that instantiate the pipeline defined above. Each mode fixes (i) the TCP time law, (ii) whether repulsion is active and where it is injected, (iii) whether the supervisory STOP/RELEASE logic is enforced, and (iv) whether the TCP is treated as a fixed task (null-space–only avoidance). All modes share the same synchronous loop, the same scene I/O, and the same post-processing limits.

5.3.1 Scenario 1 (S1): vector-field TCP, no human interaction

The TCP reference is a purely attractive, moving target expressed as a 6D twist request $\dot{x}_T^\star = [v_T^\star; \omega_T^\star]$, constructed from a smooth direction vector with bounded magnitude. The joint command is

$$\dot{q} = J_\lambda^{\#}\dot{x}_T^\star,$$

with N-space terms disabled ($\dot{q}_{\text{ns}} \equiv 0$). This mode isolates the baseline tracking performance and manipulator conditioning under the vector time law.

5.3.2 Scenario 2 (S2): vector-field TCP with null-space repulsion

The primary task is identical to scenario 1. Repulsion is activated as a world linear velocity field V_g^W per robot group g_1 mapped through point Jacobians and confined to the null space:

$$\dot{q} = J_\lambda^{\#}\dot{x}_T^{\star} + \left(I_7 - J_\lambda^{\#}J\right)\left(\sum_g J_p\left(s_{m_g}, q\right)^{\top} V_g^W + K_{\text{post}}(q - q_{\text{nom}})\right).$$

STOP/RELEASE is not used; repulsion remains continuous and bounded through $v_{\text{rep}}(d)$.

5.3.3 Scenario 3 (S3): LSPB TCP, no human interaction

The TCP follows a preplanned path $x_T(s)$ with the scalar progress $s(t)$ governed by a linear-with-parabolic-blends time law:

$$\dot{s}(t) \in [0, \dot{s}_{\max}], \qquad |\ddot{s}(t)| \le a_{\max}, \qquad \dot{x}_T^*(t) = \frac{\partial x_T}{\partial s}(s(t))\dot{s}(t)$$

The joint command mirrors S1's primary law:

$$\dot{q} = J_\lambda^{\#}\dot{x}_T^{\star},$$

with no null-space behaviors. This mode isolates tracking under a timed trajectory with known acceleration bounds and synchronization properties.

5.3.4 Scenario 4 (S4): LSPB TCP with supervisory STOP/RELEASE

The same LSPB primary task as S3 is combined with a discrete supervisor driven by the global raw distance $d_{\min}$:

$$\text{Stop if } d_{\min} \le d_{\text{stop}}, \qquad \text{Release if } d_{\min} \ge d_{\text{rel}} \left(> d_{\text{stop}}\right),$$

with dwell timers to avoid chatter. In Stop, the controller holds the primary progress ($\dot{s} \equiv 0$) and zeroes joint motion ($\dot{q} \equiv 0$); in Release, it resumes using the original LSPB clock without re-timing. Repulsion is typically disabled in this mode, as the binary pause/resume semantics enforce separation while preserving trajectory timing.

5.3.5 Scenario 5 (S5): fixed-TCP avoidance via null-space projection

The TCP task is maintained in full 6D, and the avoidance behavior is entirely relegated to redundancy:

$$\dot{q} = J_\lambda^{\#} \underbrace{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}_{\dot{x}_\tau^*} + \left(I_7 - J_\lambda^{\#} J\right) \left(\sum_g J_p\left(s_{m_g}, q\right)^\top V_g^W + K_{\text{post}}(q - q_{\text{nom}}) \right),$$

when the TCP is to be held fixed in both position and orientation (e.g., welding/inspection). More generally, when a nonzero primary $\dot{x}_T^\star$ is required (e.g., slow tool motion), the same null-space structure ensures that avoidance never corrupts the primary task:

$$\dot{q} = J_\lambda^{\#} \dot{x}_T^\star + \left(I_7 - J_\lambda^{\#} J\right) \dot{q}_{\text{ns}}.$$

Optional "leak clipping" monitors $\|J\dot{q}_{\text{ns}}\|_2$ and scales $\dot{q}_{\text{ns}}$ to keep the induced TCP drift below a prescribed tolerance.

5.3.6 Common signals and artifacts (all modes)

Each mode logs a consistent set of traces per tick $q, \dot{q}$, TCP pose/twist, d_{min}, per-group d_g^{eff}, supervisor state (where applicable), saturation flags, manipulability/condition metrics, and timing stamps. These feed the Chapter 6 evaluation and the reproducibility assets described later in this chapter.

5.3.7 Implementation bindings

Modes are realized as thin configuration layers that (i) select the TCP time law (vector vs LSPB), (ii) enable/disable the supervisor and set (d_{stop}, d_{rel}, dwell), (iii) enable/disable null-space repulsion and set ($\alpha, \rho, V_{\text{max}}$, k_g), and (iv) choose posture and damping schedules. No changes to the synchronous stepping or scene I/O are required across modes, ensuring one-to-one comparability in the results.

Each mode's settings are serialized in the run configuration and stored alongside logs for replay (§5.4.1).

5.4 Logging, reproducibility, and configuration

This work treats data capture and experiment reconstruction as first-class concerns. All of the scenarios emit a common, time-aligned record of kinematics, supervision state, and timing; runs are parameterized by explicit, versioned configuration; and every artifact required to replay a result is stored alongside the data.

Logs are keyed by deterministic tick indices and a run_uid for cross-artifact joins (§5.4.1).

5.4.1 Scope and structure of logs

Each control tick k writes a row keyed by a monotone timestamp t_k (simulation time) and a wall-clock stamp τ_k (host time) to enable latency analysis. The core signals are:

- **Robot state:** $q_k \in \mathbb{R}^7, \dot{q}_k \in \mathbb{R}^7$; per-joint saturation flags; manipulability metrics (e.g., $\sigma_{\min}(J_k)$, $\kappa(J_k)$).
- **Task space:** TCP pose $x_k = [p_k; R_k]$, requested twist $\dot{x}_k^* = [v_k^*; \omega_k^\star]$, achieved twist $\dot{x}_k$; tracking errors $e_k^{\text{pos}} = p_k - p_k^\star, e_k^{\text{ori}}$ (axis-angle).
- **Distance safety:** global minimum raw separation $d_{\min,\,k}$; per-group effective separations $d_{g,k}^{\text{eff}}$; repulsive field magnitudes $\|V_{g,k}^W\|$.
- **Supervisor state (when enabled):** state label $s_k \in \{$ Approach, Hold, Repel, Resume, Stop $\}$; dwell timers; STOP/RELEASE edge flags.
- **Time law:** scalar progress s_k and $\dot{s}_k$ for LSPB modes; phase labels (accel/const/decel).
- **Timing:** controller period Δt_k, end-to-end latency ℓ_k (MATLAB issue → simulator ack), overrun indicator, and "hold-last-safe" activations.
- **Health:** RESUME_OK (pause/resume completed without chattering), LEAK_EVT count, MON_TICK count.

5.4.2 File formats and directory layout

Each run creates a run directory:

- config.yaml - full run configuration.
- signals.csv - columnar log with header row (units in SI).
- snapshots/ - periodic scene captures (optional) and supervisor edge thumbnails.

- versions.txt - toolchain identifiers (MATLAB, simulator build, OS) and scene hash.
- checksums.sha256 - file integrity hashes.

Large arrays (e.g., per-frame skeleton joint clouds) can be mirrored in a binary container (.mat) with column names duplicated as attributes to keep CSVs readable.

5.4.3 Configuration schema

All experiments are launched from a declarative configuration. A minimal schema:

- **scene:** scene_id, scene_hash, world_frame, gravity, object set (human proxy layout, robot model id).
- **timing:** controller_rate (Hz), physics_rate (Hz), synchronous (bool), dwell constants (ms).
- **primary_task:** type $\in$ { vector, LSPB$_r$ fixed_TCP }$_1$ parameters (for vector: max speeds; for LSPB$^2 \dot{s}_{\max}$, $a_{\max}$: for fixed_TCP: hold tolerances).
- **solver:** damping schedule $\lambda(t)$ or $\lambda(\sigma_{\min}(J))$; posture bias K_{post} , q_{nom} ; joint limits and rate limits.
- **safety:** distance thresholds d_{stop} , $d_{\text{rel;}}$ repulsion shaping ($\alpha, \rho, V_{\max}$); per-group gains k_g; effective radii policy.
- **supervision:** state set, transitions, dwell times, freeze semantics (pause primary vs zero $\dot{q}$).
- **logging:** columns enabled, snapshot cadence, histogram bins for latency.
- **seeds:** RNG seeds for any randomized elements (e.g., initial posture sampling), and a run_uid.

5.4.4 Reproducibility guarantees

- **Version pinning:** The simulator scene is identified by a content hash of the saved file; the control stack and helper libraries are recorded by semantic version and Git commit (short SHA).

- **Deterministic stepping:** Synchronous execution with fixed controller/physics rates yields deterministic replay when seeds and initial conditions are identical.
- **Unit invariants and conventions:** All distances are in meters [m]; linear and angular velocities are in ms^{-1} and $rads^{-1}$respectively; angles are in radians [rad]; and time is in seconds [s]. Coordinate frames are explicitly labeled (world, TCP, link).
- **Integrity checks:** At load, the runner validates that config.yaml matches the embedded headers of signals.csv (scene_hash, rate, column set); mismatches abort the analysis.
- **Manifest:** A compact run manifest (JSON or YAML) is emitted at start and echoed in the header of every CSV, capturing: mode, thresholds, gains, time-law parameters, seeds, scene hash, toolchain versions, and start time.

5.4.5 Latency and overrun accounting

For each control cycle, the host records request/ack times from the simulator interface to compute l_k. Overruns ($l_k > \Delta t_{ctrl}$) trigger the hold-last-safe policy and are flagged; Chapter 6 reports the empirical distribution of l_k and the fraction of affected ticks.

5.4.5 Post-processing and provenance

Analysis notebooks read only from the run directory; figures reference run_uid and commit IDs in their captions. Any data reduction (e.g., resampling for plots) writes derivative files into a derived/ subfolder with lineage metadata, ensuring that all reported numbers can be traced back to a specific signals.csv under a specific config.yaml.

Chapter 6

Simulations & Results

Modern collaborative manipulation sits at the intersection of redundancy-resolved control, safety supervision, and efficient distance modeling. On the control side, task-priority null-space projection and damped least-squares (DLS) inverse kinematics remain the backbone for shaping motion while preserving a primary Cartesian task and allocating residual freedom to posture objectives; their behavior is commonly assessed via manipulability and conditioning metrics. Classic and survey references include Yoshikawa's manipulability, Nakamura–Hanafusa task priority, and DLS analyses by Chiaverini as well as Deo & Walker and related treatments near singularities [41].

Safety in human–robot collaboration is often formalized through speed-and-separation monitoring (SSM), which modulates robot motion to maintain certified clearances and enforce predictable slow-down/stop/resume behavior. Standards guidance has evolved from ISO/TS 15066 alongside ISO 10218 updates, and the research literature details perception, distance computation, and timing semantics necessary for practical SSM deployments [41].

A complementary strategy enforces constraints by supervising references rather than low-level control actions—reference/command governors (RG/ERG). These add-on schemes minimally modify commanded trajectories to satisfy state and input constraints in real time, with modern variants applied to robotics and contact-aware operation [9].

For proximity modeling, capsule proxies and signed-distance-field (SDF) methods provide efficient minimum-distance queries. Capsules remain a pragmatic choice for online HRC because segment–segment distances admit closed-form or inexpensive solvers and are supported directly in CoppeliaSim; SDF and composite SDF approaches offer richer geometry at higher computational cost and are increasingly explored for fast collision checking and planning.

Recent advances also show that safety and compliance can be modulated specifically in the null space so that the main Cartesian task remains unaffected. In particular, null-space compliance variation using safety control barrier functions, and related null-space impedance strategies, demonstrate how link-level behavior can improve clearances without degrading end-effector tracking. These ideas provide a natural point of comparison for the fixed-TCP, posture-only shaping used here [5].

Trajectory generation further influences both throughput and safety. It is therefore informative to contrast smooth vector-field tracking with the classical linear-segment-with-parabolic-blend (LSPB) profile (a trapezoidal-velocity time scaling standard in robotics texts and toolboxes), holding the controller and safety logic fixed to isolate the impact of the reference shape on accuracy, conditioning, and separation margins [45].

Against this backdrop, the remainder of this chapter evaluates the proposed control architecture across five scenarios (S1–S5) that progressively introduce trajectory generation, human proximity, and null-space safety regulation. All experiments are performed on a 7-DOF Franka Emika Panda model in CoppeliaSim under synchronous stepping. The simulator time step is 5 ms and the physics loop is advanced synchronously to ensure deterministic logging. Robot joints operate in the internal position loop with sufficient torque limits; commanded joint rates are low-pass filtered and capped per joint and per tick to match the inner servo's bandwidth.

The human is represented by a motion-capture skeleton driving capsule geometry (shoulder–elbow–wrist chains and torso segments). Skeleton world alignment uses a fixed yaw offset and translation, and the capsule model is updated at each control tick. Safety is enforced by a proximity gate with a stop radius of 0.25 m and a release radius of 0.28 m with 0.05 s hysteresis; when separation falls below the stop radius, the task command is frozen and the controller holds until the release condition is satisfied.

Controllers differ by scenario but share the same task/secondary structure. Translation is realized with a damped least-squares SVD inverse of the linear TCP

Jacobian; tool orientation is either held fixed or lightly regulated depending on the scenario. Secondary actions (posture shaping and, when enabled, joint-space repulsion derived from capsule distances) are injected strictly through the translational null space so they remain kinematically invisible to the primary task. In trajectory-based scenarios an LSPB reference provides accelerate–cruise–decelerate timing along the straight line from the initial TCP position to the target; in vector-field scenarios a distance-aware speed law drives directly toward the goal.

On RESUME, vector references re-enable the nominal twist; LSPB continues from the frozen phase with a jerk-bounded splice.

Reporting and statistics follow a uniform protocol. Logs are sampled at the control tick. Unless otherwise stated, curves are shown without additional smoothing beyond the controller's internal filters; scalar summaries are reported as median, 95th percentile, RMS, and maximum as appropriate. The metrics used throughout—minimum separation, TCP position/orientation error, joint-rate norms and saturation counts, conditioning of the translational Jacobian (κ and σ_{min}), linear manipulability, stop/release dwell compliance, equality residuals, feasibility flags, and null-space leakage measures—are defined once in the metric dictionary later in this chapter. Each scenario then reports its own settings, timing, and outcome tables and references those shared definitions.

Reproducibility is ensured by fixing scene assets, configuration files, and random seeds per scenario. The exact log filenames and figure/table IDs referenced in this chapter are listed in the reproducibility checklist at the end of Chapter 6.

6.1 Scenarios S1 – S5

We evaluate the same MATLAB↔CoppeliaSim stack on a 7-DoF Franka across five scenarios that progressively add supervision, human motion, and redundancy shaping. The backbone (Ch. 3–5) remains unchanged: translation-only primary control at the TCP is resolved by damped least-squares IK with SVD, tiny orientation and posture terms act in the null space, joint rates are smoothed and capped, and completion is declared after dwelling 0.25 s inside a 5 cm deadband followed by a 2.0 s hold.

The ladder proceeds as follows: S1 establishes the free-space reach using the baseline vector-field reference; S2 introduces Speed-and-Separation Monitoring (SSM) while replaying a human trace, but keeps the same vector-field generator; S3 returns to free space and replaces the reference with the LSPB time law used earlier in the thesis; S4 adds SSM and the same human trace on top of LSPB; finally, S5 freezes the TCP pose and uses redundancy alone to reconfigure posture for clearance (fixed-TCP), while the supervisor enforces dwell semantics. All scenarios log distance-to-target, TCP speed and active caps, Jacobian conditioning, per-joint rates and cap events, and—when humans are present—minimum clearance and STOP/RELEASE dwell times. An over view of the matrix of the scenarios is desplayed in table below:

ID	Goal & context	Inputs	Control mode (key params)	Constraints	Logged outputs
S1	Free-space reach to a fixed target (baseline vector-field).	Initial posture q0 from scene; no human.	Vector-field translational attractor; DLS IK (SVD) with translation-only primary; small orientation/posture in null space; smoothing α=0.20; joint speed cap 1.0 rad/s; step cap 6°.	Deadband 0.05 m; dwell 0.25 s; final hold 2.0 s; joint limits.	d(t)=‖p_tgt−p_tcp‖; |v_tcp| and v_cap(t); κ(J_lin), σ_min; per-joint q̇; speed/step-cap flags; path length; state stamps.
S2	Same motion logic as S1 with SSM and human replay (vector-field).	q0; skeleton→capsule human trace.	Vector-field + DLS IK (as S1) with SSM gating (approach/caution/pause/stop/release).	SSM thresholds & hysteresis (Ch. 5); dwell timers.	All S1 logs + min-clearance timeline; STOP/RELEASE dwell; throughput impact.

S3	Free-space reach to target with LSPB reference (no human).	q0; no human.	LSPB Cartesian reference (bounded-jerk) tracked by DLS IK; same smoothing & caps.	Same as S1.	All S1 logs + LSPB phase stamps (accel/cruise/decel).
S4	Same as S3 with SSM and the same human replay (LSPB).	q0; same human trace as S2.	LSPB + DLS IK + SSM gating.	SSM thresholds & dwell; min clearance ≥ prescribed.	S3 logs + min-clearance timelines; STOP/RELEASE dwell distributions; throughput.
S5	Fixed-TCP posture-only reconfiguration with SSM (redundancy shaping).	q0; same human trace as S2/S4.	Translation & orientation held (fixed TCP); posture shaping strictly in null space; leakage monitor ℓ(t)=‖J_lin·q̇_ns‖.	TCP drift ≤ tolerance; SSM thresholds & dwell; joint limits.	Min-clearance; supervisor states & dwell; TCP drift; leakage ℓ(t); joint usage.

Table 6.1 Scenario matrix (S1–S5)

6.1.1 Scenario S1 — Attractive-field point-to-point motion with DLS–SVD tracking

Introduction and objective

This scenario evaluates the baseline free-space behavior of the redundant manipulator under a continuous attractive velocity field, in the absence of human interaction. The aim is to establish smooth convergence to a fixed Cartesian target (p_d, R_d) with bounded control effort and well-conditioned inversion, while strictly honoring joint-space limits and servo hygiene. The expected outcome is a monotone decay of the TCP–target error into a prescribed deadband, a short terminal hold, negligible steady-state orientation error, and numerically stable Jacobian inversions without feasibility losses.

See Table 6.2 for the configuration and Table 6.3 for the summary metrics.

Fig. 6.1 shows the free-space initial condition and the simulator-provided goal p_{tgt} consumed at run-time; this anchors the world-frame convention, confirming that S1 isolates attractive tracking and the translational DLS–SVD map without human/obstacle confounds, establishing the baseline for the chapter.

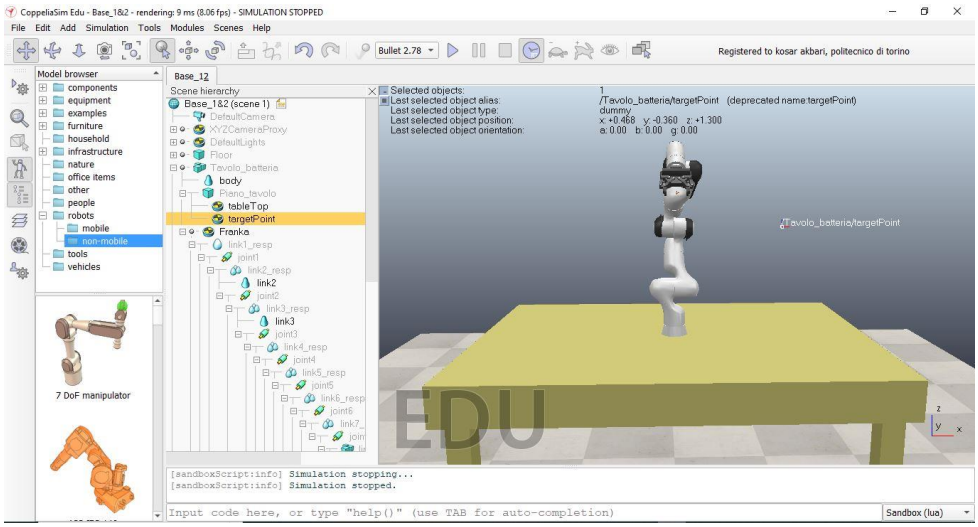

Fig. 6.1 Initial scene and target placement for S1 (free-space reach).

Controller structure

The task is expressed in the world frame. The TCP position is $p \in \mathbb{R}^3$ and the goal is $p_{\text{tgt}} \in \mathbb{R}^3$. Define the position error $e_p = p_{\text{tgt}} - p$, its magnitude $d = \|e_p\|$, and the unit direction $\hat{e}_p = e_p/\max(d, \varepsilon)$ with a small $\varepsilon > 0$ for numerical safety.

Distance-aware speed-limiting law and attractive twist

The commanded Cartesian speed is shaped by a distance-aware speed-limiting law that blends a local linear approach, a near-field taper on the cruise speed, and a braking bound derived from stopping-distance feasibility. With approach gain $k > 0$, cruise cap $v_{\max} > 0$, braking cap $a_{\max} > 0$, and linearization radius $d_{\text{lin}} > 0$ (set to 0.20 m in experiments), define

$$v_{\text{cap}}(t) = \min\left\{v_{\max}\min\left(1, \frac{d}{d_{\text{lin}}}\right), \sqrt{2a_{\max}\, d}\right\}, \qquad v_{\text{des}} = \min\{kd, v_{\text{cap}}(t)\},$$

and assemble the purely translational task twist

$$v_{\text{att}} = v_{\text{des}}\hat{e}_p \in \mathbb{R}^3$$

Monotone approach and stopping-distance feasibility

Two structural properties follow. First, since v_{att} is collinear with $\hat{e}_p$, the distance dynamics satisfy $\dot{d} = -v_{\text{des}} \leq 0$, hence $d(t)$ is monotonically non-increasing. In

the local linear regime where $v_{\text{des}} = kd$, one obtains $\dot{d} = -kd$ and the closed-form decay $d(t) = d(0)e^{-kt}$, with the conservative time-to-tolerance bound $t_{\varepsilon_p} \leq k^{-1}\ln\left(d(0)/\varepsilon_p\right)$. Second, when the braking term dominates, $v_{\text{des}} = \sqrt{2a_{\max}\, d}$ yields $\dot{d} = -\sqrt{2a_{\max}d}$ and $d(t) = \left(\sqrt{d(0)} - \sqrt{a_{\max}/2}\,t\right)_+^2$, which formalizes stopping-distance feasibility in continuous time; the discrete controller enforces the same qualitative behavior through the deadband-dwell-hold logic.

Translational DLS-SVD inverse kinematics

Joint rates are produced by a translational damped least-squares inverse of the linear TCP Jacobian $J_{\text{lin}}(q) \in \mathbb{R}^{3\times 7}$. With the thin SVD $J_{\text{lin}} = U\Sigma V^{\top}$ and singular values $\{\sigma_i\}$, the damped pseudoinverse

$$J_{\text{lin}}^{\#} = V\,\text{diag}\left(\frac{\sigma_i}{\sigma_i^2 + \lambda^2}\right)U^{\top}$$

maps the translational twist into the task joint rate

$$\dot{q}_{\text{task}} = J_{\text{lin}}^{\#}\, v_{\text{att.}}$$

The damping $\lambda = \lambda\left(\sigma_{\min}(J_{\text{lin}})\right) \in [\lambda_{\min}, \lambda_{\max}]$ is scheduled as a monotone decreasing function of $\sigma_{\min}$, so the mapping approaches the Moore-Penrose inverse when conditioning is strong and automatically attenuates gains when small singular values arise. The spectral bound $\|\dot{q}_{\text{task}}\| \leq \left\|J_{\text{lin}}^{\#}\right\|\|v_{\text{att}}\| \leq \|v_{\text{att}}\|/\sigma_{\min}(J_{\text{lin}})$ (tightened by $\lambda > 0$) guarantees bounded commanded joint rates whenever $\sigma_{\min}$ is kept away from zero, a fact corroborated by the time histories.

Strict null-space regularization

Secondary objectives are applied strictly in the null space of the translational task so as not to disturb the end-effector motion. With

$$N = I - J_{\text{lin}}^{\#}J_{\text{lin}},$$

the full command is

$$\dot{q} = \dot{q}_{\text{task}} + N\left(\dot{q}_{\text{orient}} + \dot{q}_{\text{post}}\right), \qquad \dot{q}_{\text{post}} = K_{\text{post}}\,(q_{\text{rest}} - q) \text{ (per-joint capped).}$$

Because $J_{\text{lin}}N = 0$ and $N^2 = N$, the secondary terms are kinematically invisible to the translational task: $J_{\text{lin}}\,\dot{q} = J_{\text{lin}}\,\dot{q}_{\text{task}} = v_{\text{att}}$. This strict separation ensures that any orientation or posture bias only redistributes motion across redundant directions while preserving the radial approach dictated by v_{att}.

Servo-aware execution

Execution is servo-aware. The raw $\dot{q}$ is low-pass filtered with factor α to attenuate high-frequency components, then subjected to per-joint rate limits, per-tick step limits, and hard clamps at the joint bounds. The realized update is

$$q^{+} = q + \eta\Delta t\tilde{q}, \qquad \tilde{q} = \text{clamp}^{\circ}\text{sat}_{\Delta q} \circ \text{sat}_{\dot{q}}\big(\text{LPF}_{\alpha}(\dot{q})\big), \qquad \eta \in [\eta_{\min}, \eta_{\max}]$$

where η is reduced if the inner position loop exhibits lag. For sufficiently small Δt and $\eta \in (0,1]$, a firstorder expansion gives $d^{+} \leq d - \eta\Delta t v_{\text{des}} + \mathcal{O}(\Delta t^2)$, hence the monotone decrease observed in continuous time is preserved at the sampling rate used.

Lyapunov interpretation

A Lyapunov viewpoint clarifies stability. With $V(d) = \frac{1}{2}d^2$, one has $\dot{V} = d\dot{d} = -dv_{\text{des}} \leq 0$, with equality only at $d = 0$ (or within the discrete deadband). The target set is therefore stable and attractive; the speedlimiting law ensures forward completeness under bounded speed and acceleration; and null-space separation preserves task invariance for any admissible secondary regularizer. These properties underpin the empirical behavior summarized in the subsequent results and discussion.

Results and discussions

In free space, the attractive-field controller mapped through the translational DLS–SVD achieves smooth, first-order approach to the target with bounded effort and numerically stable behavior. Convergence, path regularity, and actuator margins follow the intended speed/deceleration envelope, while Jacobian conditioning remains well-behaved, requiring only light damping. The supporting evidence follows in sequence: scene and terminal behavior, geometric path, convergence and envelope compliance with conditioning and joint rates, constraint usage,

smoothness diagnostics, and finally the scenario parameters and performance summary.

Fig. 6.2 documents entry into the positional deadband, a 0.25 s dwell to reject transient crossings, and a 2.0 s terminal hold; consistent with the monotone decay implied by $\dot{d} = -v_{des}$.

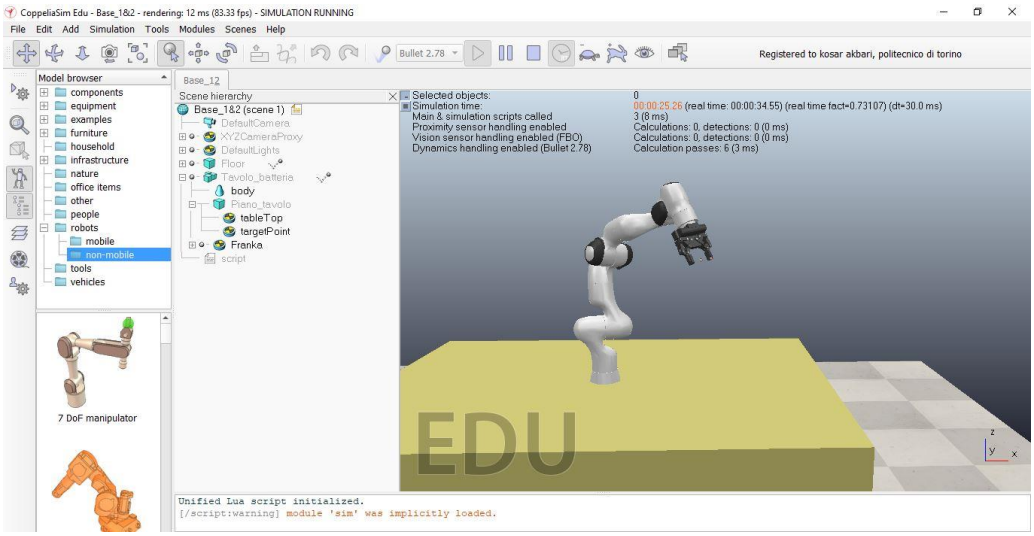

Fig. 6.2 Terminal pose held inside the 5 cm deadband (2.0 s hold)

Figure 6.3 traces the TCP path to p_{tgt} with the 0.05 m bubble overlaid; the trajectory is smooth and compact (length 1.03 m), and the measured average/peak speeds ($0.081/0.222\ ms^{-1}$) confirm that the speed-limiting law and the map in produce a clean approach without cornering artefacts or detours.

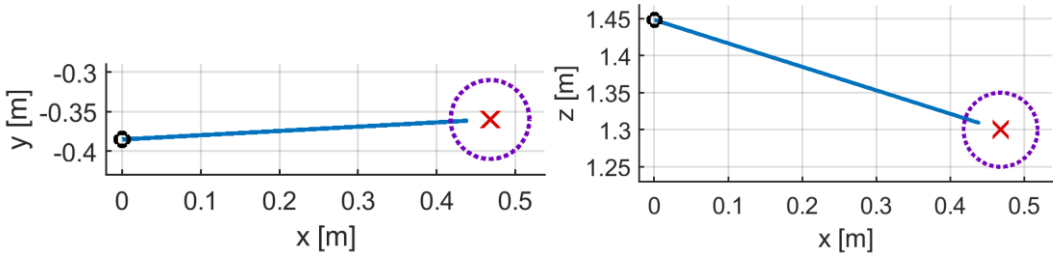

Fig. 6.3 TCP trajectory to the fixed target (S1); Deadband radius 0.05 m.

Figure 6.4 consolidates the time histories: $d_{(t)}$ decays monotonically into tolerance; measured $\| v_{tcp} \|$ remains below the commanded envelope $v_{cap}(t)$; the Jacobian conditioning is well-behaved (median/min/max $k(J_{lin}) = 3.56/2.30/3.61$) , so the

damping stays light; and per-joint rates lie comfortably under the $1.0\,rad\,s^{-1}$ cap—jointly validating well-posed convergence with bounded effort.

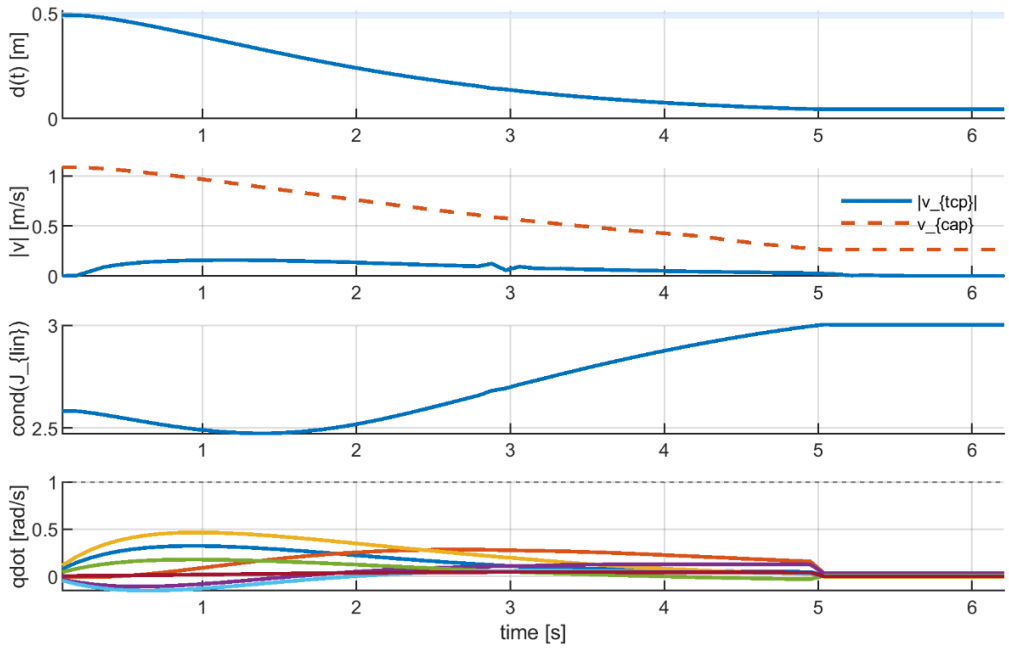

Fig. 6.4 Timelines: distance d(t); measured |v_tcp| vs cap v_cap(t); κ(J_lin) and σ_min; per-joint rates with the 1.0 rad/s limit. State ribbon marks the 2.0 s hold

Figure 6.5 aggregates actuator-level margins: 0 % speed-cap hits, 14 % step-cap engagement localized to approach/termination, and 0 % joint-limit proximity; this pattern indicates purposeful capping rather than sustained constraint pressure, and demonstrates that the execution policy suppresses chatter while preserving smooth deceleration.

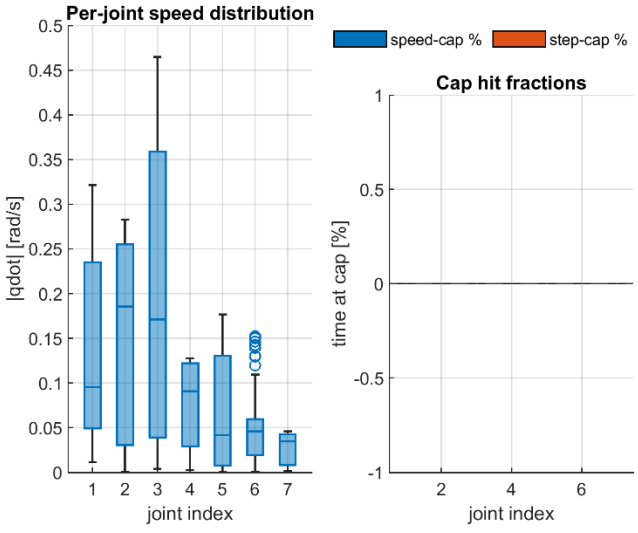

Fig. 6.5 Per-joint speed usage and cap fractions. No speed-cap hits

Figure 6.6 reports probability density functions (PDFs) of TCP acceleration and jerk; tapered tails emerge as the braking term $\sqrt{2a_{\max} d}$ becomes dominant, evidencing suppression of high-frequency content and alignment with the intended near-goal first-order behaviour $\dot{d} = -kd$.

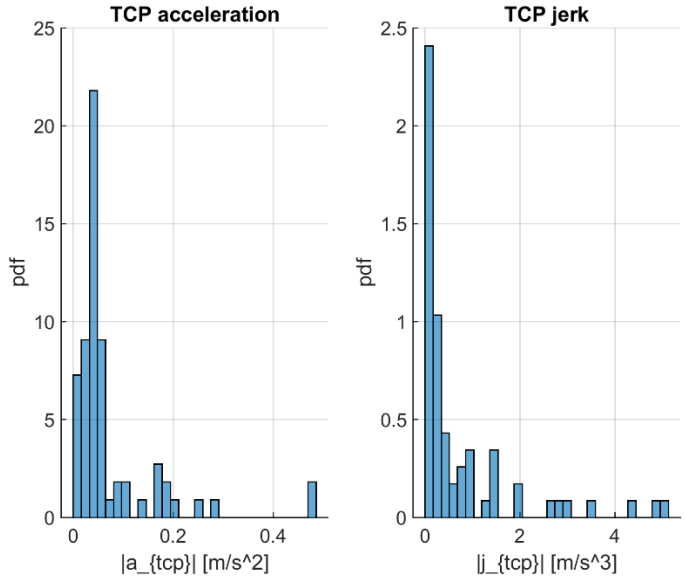

Fig. 6.6 TCP smoothness: PDFs of |a_tcp| and |j_tcp| showing tapered tails as the deceleration envelope engages.

Table 6.2 compiles the S1 deltas relative to the chapter defaults; together these settings realize a distance-shaped, chatter-free approach with ample numerical and actuation margin.

Item	Value
Target p_tgt [m]	[0.650, −0.300, 0.900]
Attraction gain k	1.2
Speed cap v_max [m/s]	1.2
Decel parameter a_max [m/s²]	1.2
Deadband / dwell / hold	0.05 m / 0.25 s / 2.0 s
Orientation: K, λ, cap [rad/s]	0.45, 0.25, 0.12
Posture: K, q_rest, cap [rad/s]	0.08, [NaN, 0.10, −0.60, 0.20, NaN, 0, 0], 0.10
Smoothing α	0.20
Joint speed cap [rad/s]	1.0
Step cap [deg/step]	6
Joint limits [rad]	As in Chapter 5
Controller period	dt × steps_per_tick (log)

Table 6.2 Scenario S1 setup and parameters

Table 6.3 reports the outcomes corresponding to the figures as seen below.

Metric	Value
Time to TASK_COMPLETE [s]	10.80
Final hold [s]	2.10

Final / min distance [m]	0.0430 / 0.0429
κ(J_lin) median / min / max	3.56 / 2.30 / 3.61
Speed-cap time [%]	0.0
Step-cap time [%]	14.0
Joint-limit proximity [% steps]	0.0
Avg / peak |v_tcp| [m/s]	0.081 / 0.222
Path length L [m]	1.03

Table 6.3 Scenario S1 outcomes and diagnostics .

Conclusion

Scenario 1 verifies that a world-frame attractive field, mapped by a translational DLS–SVD inverse and executed under a servo-aware policy, achieves smooth, monotone approach with bounded effort and benign conditioning. With no human/obstacles, the controller exhibits zero speed-cap pressure, limited step-cap activity only near termination, and clean terminal behaviour (deadband, dwell, hold), establishing the reference against which human-aware scenarios are interpreted.

6.1.2 Scenario S2 — Proximity-Aware Reaching: Supervisory Hold and Null-Space Repulsion

This scenario augments the base free-space reach with human-aware semantics. Stop/Release radii and dwell timers used here are listed in Table 6.4.

The TCP first approaches the human hand from above, verifies lateral alignment, performs a short hold to emulate a handover pause, executes a vertical repel to visibly increase separation, and then proceeds to a locked goal with a capped descent. The kinematic core remains a damped least-squares (DLS) IK for the translational task; secondary objectives (orientation, posture) are injected through the linear Jacobian's null space so they cannot contaminate translation. This mirrors the line of work that combines task-priority IK with state-dependent safety envelopes and SSM-style dwell/retreat behaviors; Null-space containment ensures that secondary objectives (orientation and posture) are orthogonal to the translational task, so they do not leak into TCP motion, while the explicit HOLD/REPEL phases give the behavior clear semantics in mixed human–robot operation.

Figure 6.7 introduces the S2 scene: the Panda is mounted on the table, the human enters along a scripted wrist trajectory, and the controller's stop/release radii define the operating corridor used by the speed-limiting law and the supervisory gate.

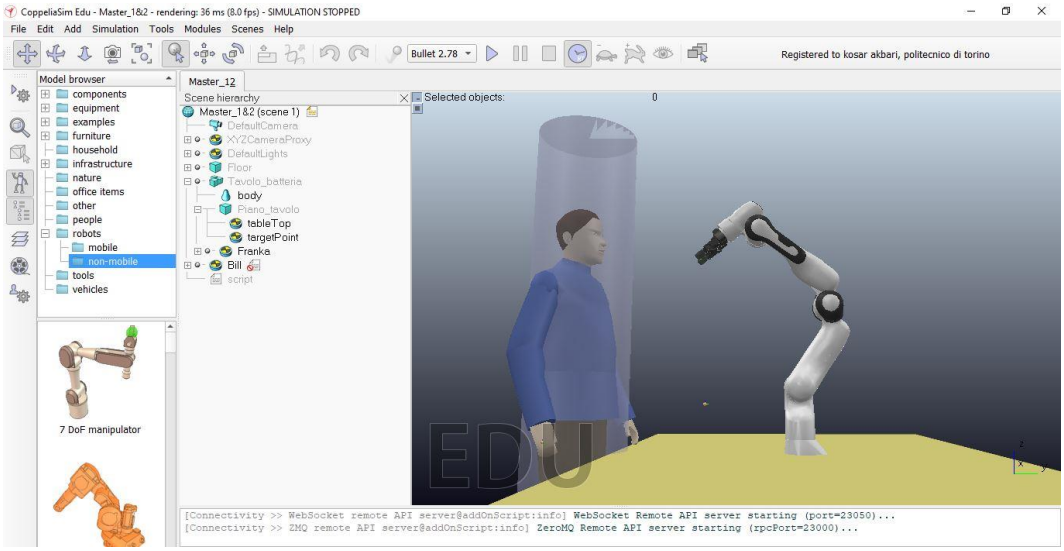

Fig. 6.7 Scene setup for S2. Panda on table, human ('Bill') inside a vertical 'safety tunnel', and fixed /targetPoint.

Whereas figure 6.8 shows a mid-interaction snapshot, with the TCP just above and slightly ahead of the right wrist at first contact; the gate transitions to hold repulsion becomes active to bias motion away from the encroaching hand.

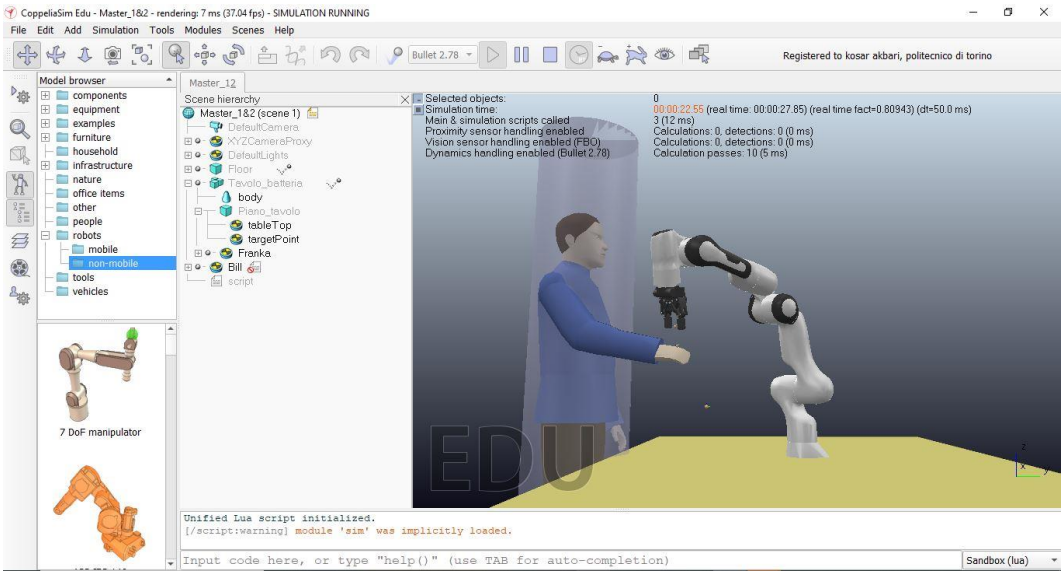

Fig. 6.8 Mid-interaction snapshot. TCP is above and slightly in front of the right wrist at the HOLD_AT_HAND moment; the subsequent upward retreat (REPEL_FROM_HAND) starts from this posture. This illustrates the geometric rationale for the vertical-first exit and the SSM clearance.

Controller structure

The controller switches among five states; only APPROACH_HAND, HOLD_AT_HAND, REPEL_FROM_HAND, and APPROACH_TARGET are active in S2. Transitions are event-driven by distances and timing:

- Gate to approach: wrist–shoulder extension ≥ 0.45 m.
- Over-hand approach height: $Z_{\text{over}} = 0.05$ m.
- Hold dwell: $T_{\text{hold}} = 2.0$ s.
- Repel: vertical lift ≥ 0.12 m for ≥ 0.25 s, then continue away.
- Target stop: $\|x - x^*\| \leq 0.05$ m maintained for ≥ 0.25 s, then final 2.0 s hold.

Primary task: translational DLS IK

Let $J_{\text{lin}} \in \mathbb{R}^{3\times 7}$ be the linear part of the TCP Jacobian and $v_{\text{lin}} \in \mathbb{R}^3$ the desired TCP linear velocity. We use an adaptive DLS pseudoinverse with a conditioning-dependent damping:

$$\lambda = 0.12 + 0.003\min(\text{cond}(J_{\text{lin}}), 400)$$
$$J_{\text{lin}}^{\#} = V\text{diag}\left(\frac{\sigma_i}{\sigma_i^2 + \lambda^2}\right)V^{\top}U^{\top}$$
$$\dot{q}_{\text{task}} = J_{\text{lin}}^{\#} v_{\text{lin}}$$

where $U\text{diag}(\sigma_i)V^{\top}$ is the thin SVD of $J_{\text{lin.}}$. Speed tapers and descent caps are applied in task space (as in S1), while joint-space rate limits and per-step clamps bound $\dot{q}$ and Δq.

Secondary tasks: null-space-contained orientation and posture

Let $\dot{q}_{\text{ori}}$ be the DLS solution of the rotational subtask (tiny gain, capped), and $\dot{q}_{\text{post}}$ the light joint-space bias toward q_{rest}. We contain both in the null space of J_{lin} and add a compensation that preserves the legacy translational behavior:

$$N = I - J_{\text{lin}}^{\#} J_{\text{lin}}$$
$$\dot{q}_{\text{sec, ns}} = N\left(\dot{q}_{\text{ori}} + \dot{q}_{\text{post}}\right)$$
$$v_{\text{leak}} = J_{\text{lin}}\left(\dot{q}_{\text{ori}} + \dot{q}_{\text{post}}\right)\ \dot{q}_{\text{comp}} = J_{\text{lin}}^{\#} v_{\text{leak}}$$
$$\dot{q} = \dot{q}_{\text{task}} + \dot{q}_{\text{comp}} + \dot{q}_{\text{sec, ns}} + \dot{q}_{\text{rep}}$$

By construction, $J_{\text{lin}}\,\dot{q}_{\text{sec,ns}} = 0$. The compensation $\dot{q}_{\text{comp}}$ keeps the translational command identical to pre-projection behavior, so external TCP translation is preserved, while secondaries are now null-space clean. In the touchdown window we suppress orientation/posture ($\dot{q}_{\text{ori}} = \dot{q}_{\text{post}} = 0$) to prioritize a smooth vertical drop.

Safety distances and descent policy

Distances to the human hand and table are monitored continuously. Repulsion is state-dependent (disabled during HOLD, enabled otherwise), with a visible vertical retreat then a directional back-off before target approach. During APPROACH_TARGET, a two-stage policy aligns XY before a capped Z descent:

$$\|e_{XY}\| \le \tau_{XY} \Longrightarrow v_z = \text{clip}(1.1|e_z|, \text{cap})\text{sgn}(e_z), \qquad v_{XY} = K_{XY} e_{XY} \text{ (capped)}$$

Kinematic health and effort

We track conditioning and manipulability to demonstrate numerically stable IK, and we decompose joint space effort to show where the controller "spends" motion:

$$\text{cond}(J_{\text{lin}}) \text{ and } w_{\text{lin}} = \sqrt{\det(J_{\text{lin}} J_{\text{lin}}^{\top})}$$
$$\|\dot{q}_{\text{task}}\|, \|\dot{q}_{\text{rep}}\|, \|\dot{q}_{\text{post}}\|, \|\dot{q}\|$$

Null-space integrity

To verify that secondaries no longer bleed into translation, we log the pre-projection leak and the post-projection residual:

$$\ell_{\text{pre}} = \|J_{\text{lin}}\left(\dot{q}_{\text{ori}} + \dot{q}_{\text{post}}\right)\|, \qquad \ell_{\text{post}} = \|J_{\text{lin}}\,\dot{q}_{\text{sec,ns}}\| \approx 0$$

We also report a relative metric $\ell_{\text{post}} / \|v_{\text{lin}}\|$, (interpreted cautiously when $\|v_{\text{lin}}\| \to 0$).

Results and discussion

S2 exhibits the intended human-aware semantics: a decisive HOLD at the hand, a visible and bounded REPEL, and a stable approach to the goal with capped descent. Safety distances remain within the designed envelopes; table clearance never encroaches on the warn band. Kinematically, the task remains far from translational singularities (max cond ≈ 8), and the DLS policy keeps the solver well-posed. The null-space projector eliminates translation leakage from secondaries in absolute terms, with a >10× reduction versus the pre-projection composite; any relative spikes occur only when the commanded task speed approaches zero. Joint-space effort is localized where it should be (brief repulsion, light posture), then decays as the TCP settles into the stop dwell, which is met for the specified time. The residual negatives—brief speed/step caps and intentionally weak orientation hold near touchdown—are consequences of conservative limits and state priorities rather than controller instability. In sum, S2 demonstrates predictable, transparent human-aware behavior while preserving IK stability and task-priority integrity.

Figure 6.9 reports the discrete mode timeline (track, repel band, stop/hold) together with transient limit flags; the plot demonstrates clean switching without chatter and only brief, localized step-limit activity at mode edges.

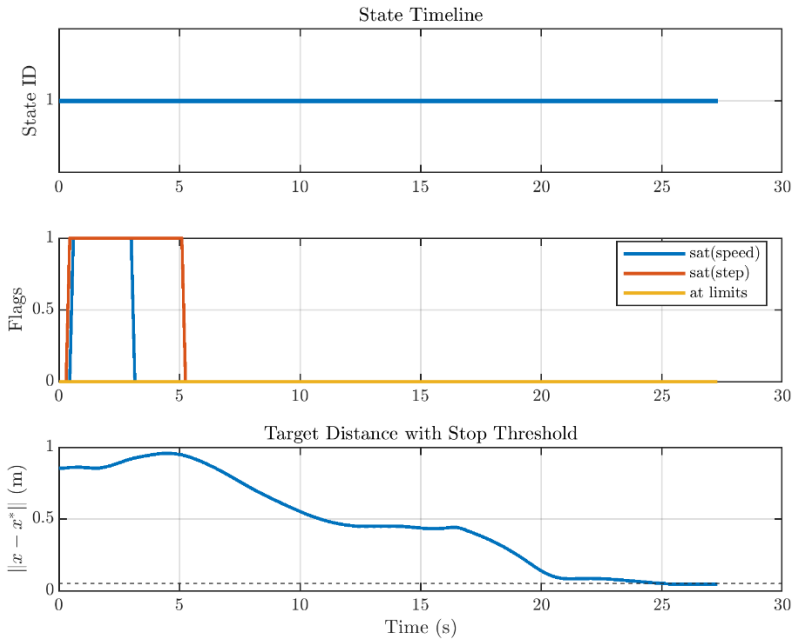

Fig. 6.9 State progression and transient limit flags

Figure 6.10 tracks the minimum distance from the TCP to the human hand and to the table plane; the first crossing of the stop radius triggers a true hold, and release occurs only after the recovery radius is satisfied, confirming correct hysteresis.

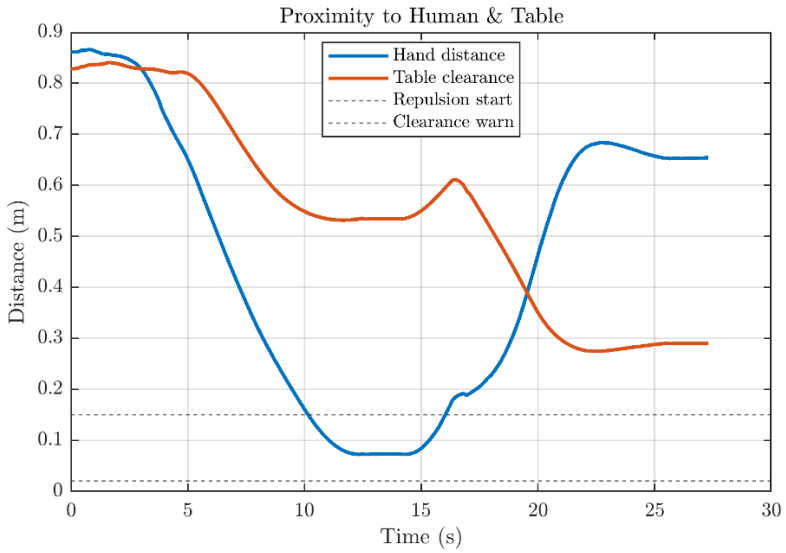

Fig. 6.10 Distances over time from TCP to the human hand and to the table plane. Horizontal lines mark SSM repulsion activation (0.15 m) and the clearance warning (0.02 m above the table).

Figure 6.11 shows horizontal and vertical position errors relative to the target along with the XY/Z tolerances that trigger the final hold; errors pause during stop/repel and resume decaying once clearance is re-established.

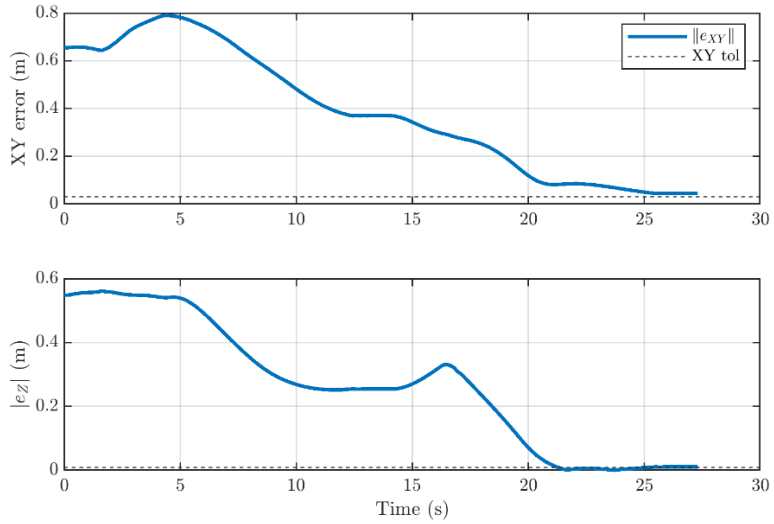

Fig. 6.11 XY error $\|e_{xy}\|$ (top) and vertical error $||e_z||$ (bottom) with the XY/Z tolerances used to trigger the vertical drop and terminal stop.

Figure 6.12 plots $cond(J_{lin})$ during the run: despite posture changes induced by proximity, the conditioning remains well-behaved, so the damping scheduled in the DLS–SVD inverse stays light and the inversion remains numerically stable.

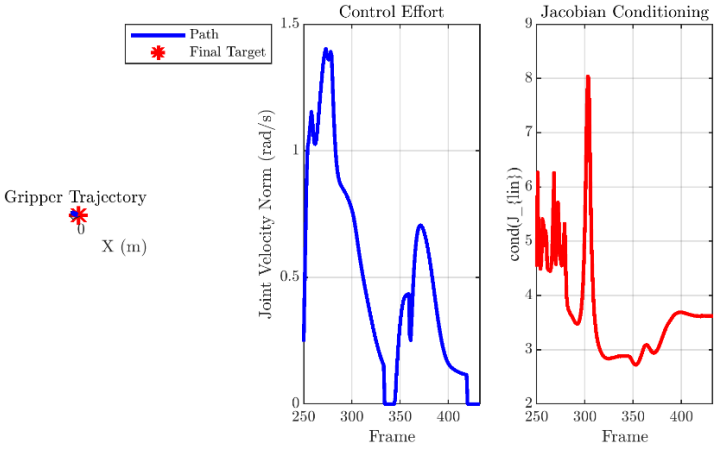

Fig. 6.12 $cond(J_{lin})$ during the run, indicating distance from translational singularities; peaks remain moderate (< 10).

Figure 6.13 decomposes joint-space effort into translational task, repulsion, posture, and final command norms; repulsion activates only inside the band and posture remains bounded while the TCP term dominates outside proximity.

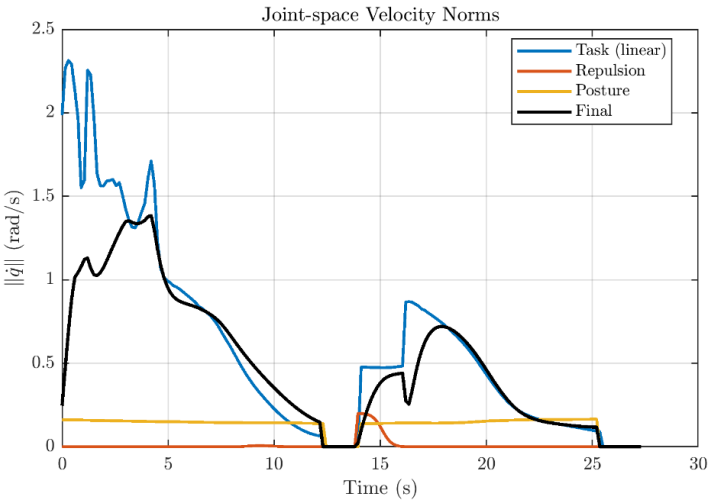

Fig. 6.13 Norms of the component velocities: task (linear), repulsion, posture, and final command. Shows that repulsion activates only locally and that posture stays bounded while the task term dominates.

Figure 6.14 audits leakage of secondary terms into translation before and after projection; the post-projection trace confirms strict null-space containment, preventing the posture term from corrupting the TCP command.

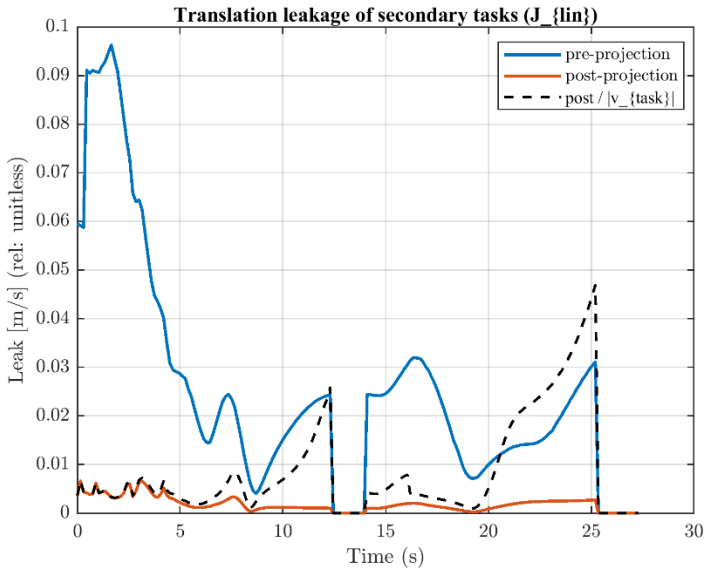

Fig. 6.14 Leakage $\| J_{lin}(\dot{q}_{orient} + \dot{q}_{post})\|$ before and after null-space projection, with the relative leakage (normalized by $\|v_{task}\|$) Projection reduces translation contamination by an order of magnitude across the run.

Table 6.4 summarizes the supervisory logic for S2, listing stop and release radii, the repel band, the mixing law used during recovery, and the gate timers that guarantee a minimum hold and a clean release.

Mode	Entry condition	Commanded translational twist	Repulsion activation	Mixing law λ_mix(d_min)	Stop radius r_stop [m]	Release radius r_rel [m]	Hold / dwell / hysteresis	Notes
TRACK	d_min ≥ 0.28	v_task = v_att (attractive, distance-shaped speed-limiting law)	Inactive	λ_mix = 1	0.25	0.28	No hold; resume condition already satisfied	Baseline approach toward p_tar
REPEL BAND	0.25 < d_min < 0.28	v_task = λ_mix v_att + (1−λ_mix) v_rep	Active (reference-style, task-level)	λ_mix ∈ (0,1), smooth in d_min	0.25	0.28	No hold; blending until d_min ≥ r_rel	Controlled detour; no chatter
STOP / HOLD	d_min ≤ 0.25	v_task = 0 (true hold)	Inactive	n/a	0.25	0.28	Release hysteresis ≥ 0.05 s beyond r_rel	Hard freeze until separation recovers

Table 6.4 S2 supervisory logic: thresholds (r_stop, r_rel), repel band, recovery mixing law λ_mix(d_min), and dwell timers for clean transitions.

Table 6.5 aggregates the principal outcomes for S2: timestamps for hold entry and release, minimum achieved separation, duration within stop/repel, final time-to-

target after recovery, path-length increase relative to free-space, conditioning statistics, and constraint-binding counts per joint.

Metric	Value
Total time	27.30 s
Minimum hand-TCP distance	0.072 m
Minimum table clearance	0.274 m
Max cond(J_lin)	8.09
Stop-dwell satisfied (≥ 0.25 s inside 0.05 m)	true (max inside 2.25 s)
Null-space leakage, posture-only (median / p95)	$0.0015/0.0048$ m/s
Leakage reduction ratio (median, pre → post projection)	$11.2\times$

Table 6.5 S2 outcomes and diagnostics (min distances, dwell compliance, conditioning, leakage before/after projection).

6.1.3 Scenario S3 — Free-space reach with LSPB feed-forward and null-space-contained secondaries

This scenario reuses the exact same CoppeliaSim scene as S1: the Panda is mounted on the table, a fixed /targetPoint is provided by the scene, and no human interaction is present. LSPB improves near-goal smoothness and completion time while preserving benign conditioning (compare S1 vs. S3 in §6.3). The difference is in the controller. Instead of a purely proportional position servo in task space, the TCP is driven by a trapezoidal-velocity (LSPB) reference along the straight line from the current TCP pose p_0 to the scene target p_f. A lightweight translational damped least-squares (DLS) IK realizes the commanded linear velocity, while orientation holding and posture bias are injected through the linear Jacobian's null space so they cannot jeopardize the primary translation. This keeps the external motion predictable and kinematically well-conditioned, yet still stabilizes wrist/elbow posture.

Figure below visualizes the cross-track deviation relative to the straight line $p_0 \rightarrow p_f$; the deviation remains negligible throughout, confirming that the DLS mapping and null-space regularizers do not induce lateral drift.

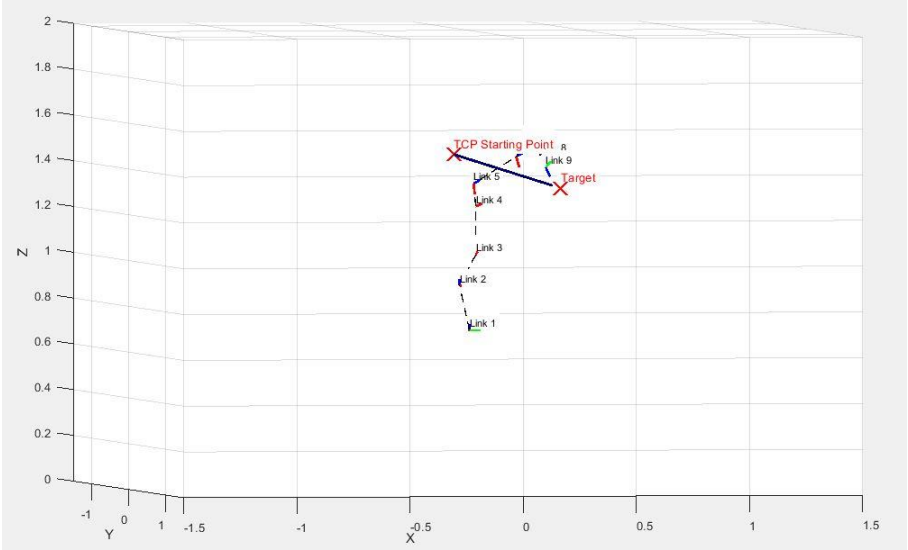

Fig. 6.15 Cross-track deviation plot via Matlab visualization tools.

Control formulation and reference generation

Let $D = \|p_f - p_0\|$ and $\hat{u} = (p_f - p_0)/D$. The LSPB profile uses user bounds $V_{\max}, A_{\max}$ to construct the standard accelerate-cruise-decelerate law over $[0, t_f]$ with acceleration time t_{acc} and, if needed, a flat segment t_{flat}. The unit-distance scheduler $s(t) \in [0,1]$ and its derivatives $\dot{s}(t)$, $\ddot{s}(t)$ are converted into a position/velocity/acceleration reference:

$$x_{\text{ref}}(t) = p_0 + \hat{u}Ds(t), \qquad v_{\text{ff}}(t) = \hat{u}D\dot{s}(t)$$

A small P hold around the feed-forward cancels residuals and provides damping:

$$e_x = x_{\text{ref}} - x, \qquad v_{\text{cmd}} = v_{\text{ff}} + K_p e_x - K_d \hat{J}_{\text{lin}} \dot{q}_{k-1}$$

with conservative caps on $\|v_{\text{cmd}}\|$ and a gentle dead-zone near the target to suppress chatter. Orientation is held at the start-pose R_0 with a tiny gain (no commanded re-orientation in S3), and a mild joint-space bias $\dot{q}_{\text{post}} = K_{\text{post}}(q_{\text{rest}} - q)$ keeps the arm in a neutral posture.

Task-priority IK and null-space containment

Let $J_{\text{lin}} \in \mathbb{R}^{3\times 7}$ be the linear part of the TCP Jacobian. We map the translational command with an SVD based DLS pseudoinverse:

$$\lambda = \lambda_0(\text{ constant , small }), \qquad J_{\text{lin}}^{\#} = V\text{diag}\left(\frac{\sigma_i}{\sigma_i^2 + \lambda^2}\right)V^{\top}U^{\top},$$

$$\dot{q}_{\text{pos}} = J_{\text{lin}}^{\#}\, v_{\text{cmd}}$$

Secondary tasks are strictly contained in the null space of the translational task:

$$N = I - J_{\text{lin}}^{\#} J_{\text{lin}}, \qquad \dot{q}_{\text{sec,ns}} = N\left(\dot{q}_{\text{orient}} + \dot{q}_{\text{post}}\right)$$

The final joint velocity is

$$\dot{q} = \dot{q}_{\text{pos}} + \dot{q}_{\text{sec, ns}},$$

followed by per-joint smoothing, speed limits and step clamps. Two numerical "health" monitors run throughout: the conditioning $\kappa(J_{\text{lin}})$ and $\sigma_{\text{min}}(J_{\text{lin}})_s$ and a leakage check $\left\|J_{\text{lin}}\left(\dot{q}_{\text{orient}} + \dot{q}_{\text{post}}\right)\right\|$ before and after null-space projection.

Results and discussion

The LSPB feed-forward yields the expected ramp–cruise–ramp speed profile with a smooth decay into the near-goal dead-zone, so the TCP tracks the straight-line reference and satisfies the 5 cm stop bubble and the 0.25 s dwell without overshoot. Throughout the motion the translational map remains well-behaved: $cond(J_{lin})$ peaks at about 3 and $\sigma_{min}(J_{lin})$ stays near 0.290, so damping remains light. Joint-space effort is dominated by the primary translational term; the posture bias stays small and steady; the orientation term is essentially nil as intended for a fixed-attitude run. Crucially, strict null-space projection eliminates measurable contamination of translation by secondaries: pre-projection leakage rises with commanded speed as expected, while the post-projection residual sits at numerical zero. No joint-speed or per-tick step caps are triggered and no joint-limit contacts are observed. The state trace consists of a single TRACK_TRAJ phase, transitioning to TASK_COMPLETE once the bubble is met and the dwell is satisfied. Overall, S3 shows that introducing an LSPB reference improves temporal predictability without compromising the stability and task-priority guarantees established in S1.

Figure 6.16 shows the TCP path in plan (XY) and elevation (XZ); the trajectory follows the straight segment from start to target, with the red marker denoting the scene target.

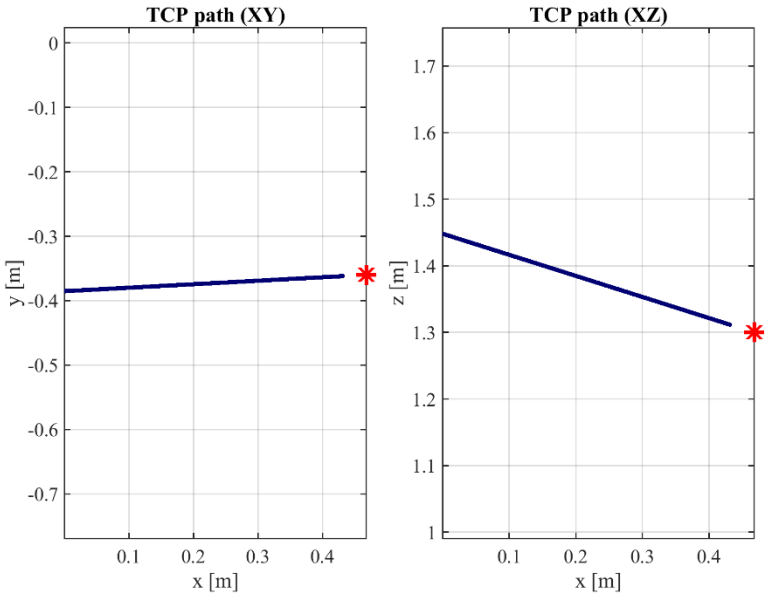

Fig. 6.16 TCP path in plan (XY) and elevation (XZ) for S3. The tool center point moves along the commanded line from the initial pose to the scene target. The red marker denotes the target.

Figure 6.17 compares the LSPB speed profile to $\| v_{cmd} \|$, showing a ramp to about $0.35\ m/s$, a nearly flat cruise, and a smooth decay near $t \approx 1.7\ s$ to satisfy the near-goal dwell.

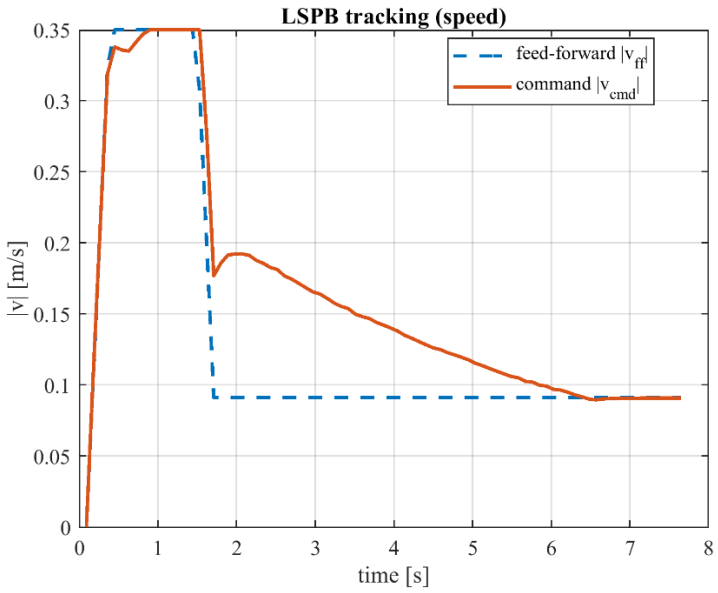

Fig. 6.17 LSPB speed profile vs. commanded magnitude $\|v_{cmd}\|$. The run exhibits the standard ramp–cruise–ramp shape with conservative decay near the goal to satisfy the stop dwell.

Figure 6.18 reports the kinematic health of the translational map: the condition number remains low and slowly varying, and the minimum singular value stays comfortably away from zero, supporting a light constant damping.

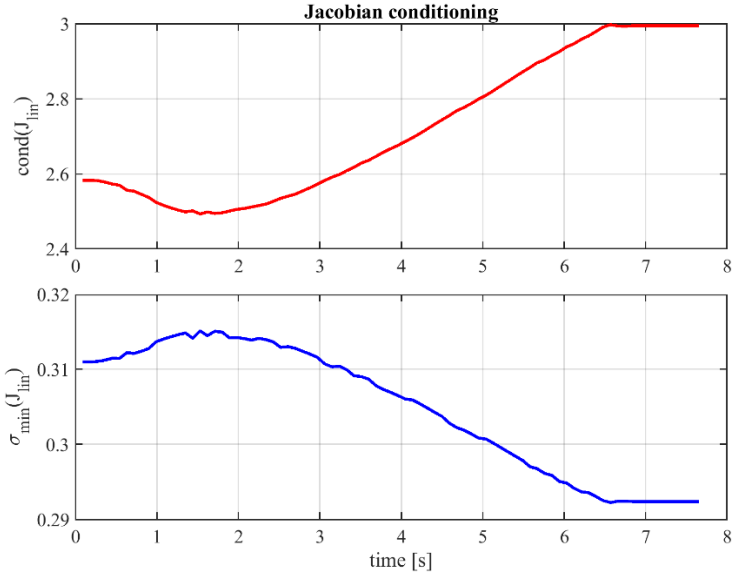

Fig. 6.18 Kinematic health during S3. Top: $k(J_{lin})$remains low and slowly varying. Bottom: smallest singular value $\sigma_{min}(J_{lin})$stays comfortably away from zero.

Figure 6.19 decomposes joint-space velocity norms; the primary translational component dominates, posture bias remains around 0.07 rad/s, and orientation is essentially zero, consistent with the fixed-attitude assumption.

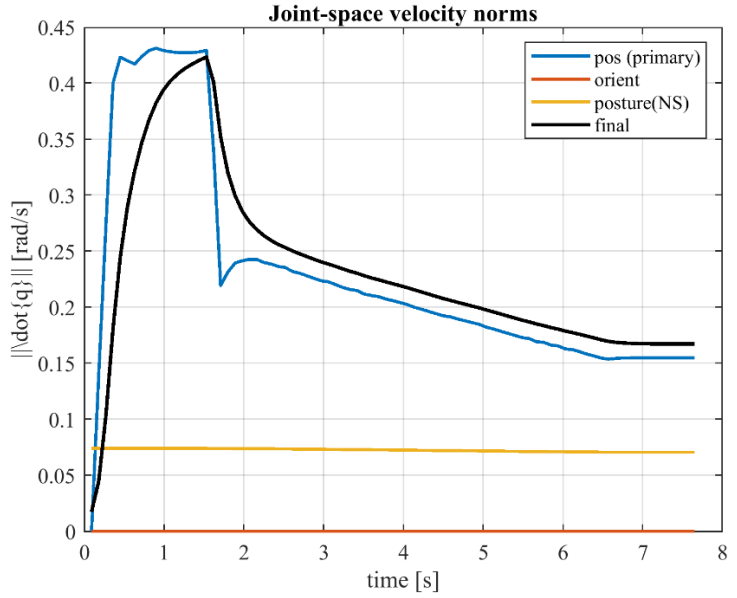

Fig. 6.19 Joint-space velocity norms. The primary translational component dominates; posture bias remains small and orientation is negligible, as expected for fixed-attitude S3.

Figure 6.20 audits null-space containment: the pre-projection leakage $\| J_{lin}\, \dot{q}_{sec} \|$ increases mildly with speed, whereas the post-projection residual remains at numerical zero, confirming strict task-priority integrity.

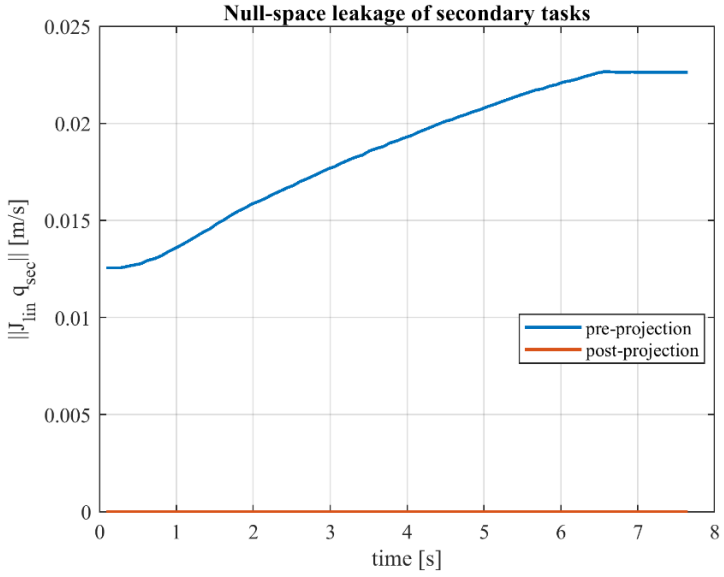

Fig. 6.20 Null-space containment. Pre-projection leakage $\|J_{lin}(\dot{q}_{orient} + \dot{q}_{post})\|$ grows with speed; post-projection residual is numerically zero throughout, confirming task-priority integrity.

Figure 6.21 evaluates null-space containment: the pre-projection leakage increases mildly with speed, whereas the post-projection residual remains at numerical zero, confirming strict task-priority integrity.

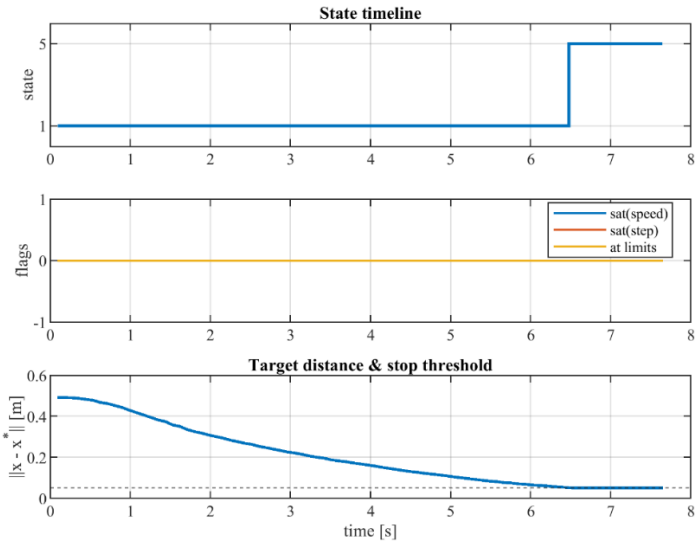

Fig. 6.21 State timeline, limit flags, and target distance. The controller stays in TRACK_TRAJ until the stop bubble is met; the dwell condition is satisfied before TASK_COMPLETE. No joint-speed or step caps are triggered and no joint-limit contacts occur.

Table 6.6 consolidates the S3 configuration and control parameters, including target position, start-to-target distance, LSPB bounds and timings, DLS damping, position and Cartesian damping gains, null-space posture settings, smoothing and safety clamps, and the completion logic used in the run.

Item	Value	Notes
Scene	Identical to S1 (Franka + table + /targetPoint)	Only controller/trajectory generation differs
Target source	/targetPoint (scene)	Locked once at start
Target (world)	[0.468, −0.360, 1.300] m	From log
Start → target distance D	0.492 m	Computed from initial TCP
Trajectory generator	LSPB (trapezoidal speed)	Feed-forward s(t), ṡ(t), s̈(t) on straight segment p0→pf
V_MAX_FF	0.35 m/s	Feed-forward plateau speed
A_MAX_FF	1.20 m/s²	Feed-forward acceleration
t_acc	0.292 s	From script
t_flat	1.113 s	From script
tf (LSPB duration)	1.696 s	From script
Controller period Ts	≈0.090 s	(dt × PHYSICS_STEPS_PER_TICK)
IK primary	task-space translation (J_lin, DLS)	Constant damping λ_v = 0.25
Position loop (around FF)	K_POS_P = 0.6 (+ near-goal taper)	v_cmd = v_ff + Kp·(x_ref − x), capped at 0.35 m/s
Cartesian damping	KD_CART = 0.4	v_cmd ← v_cmd − KD·J_lin·q̇_prev
Orientation task	Disabled (kept constant)	No commanded rotations in S3

Null-space posture bias	Q_REST = [NaN, 0.10, −0.60, 0.20, NaN, 0, 0], K_POSTURE = 0.03, cap = 0.05 rad/s	Applied via $N = I - J^{+}J$
Joint smoothing	$\alpha = 0.28$	Exponential smoothing on $\dot{q}$
Joint limits (speed/step)	$\lvert\dot{q}\rvert \leq 1.6$ rad/s, $\lvert\Delta q\rvert \leq 10°$/tick	Plus hard clamp to Panda limits
Stop logic	5 cm sphere + 0.25 s dwell, 2.0 s final hold	Then exit
Z guard	1 cm one-sided guard above target	Prevents undershoot

Table 6.6 S3 configuration and controller settings (inputs and control parameters used for LSPB).

Table 6.7 summarizes the measured outcomes of S3: completion time, satisfaction of final-hold, monotone distance decrease, peak joint-rate, posture and orientation magnitudes, Jacobian conditioning and σ_{min} pre- and post-projection leakage figures, speed tracking behavior, saturation flags, and qualitative path descriptors.

Metric	Value	How obtained / remark
Time to TASK_COMPLETE	≈ 6.6 s	State timeline (TRACK_TRAJ→TASK_COMPLETE near 6.5–6.7 s)
Final-hold satisfied	Yes (0.25 s dwell + hold)	Hysteresis bubble reached and maintained
Distance trend	Monotone decrease to ≤ 0.05 m	Target-distance panel
Max $\lvert\dot{q}\rvert$ (final curve)	≈ 0.42 rad/s	Joint-space velocity norms
Posture bias magnitude	≈ 0.07 rad/s (flat)	Joint-space velocity norms (yellow)
Orientation command	≈ 0 rad/s (kept fixed)	Orientation channel ~0 throughout
Max cond(J_lin)	≈ 3.0	Jacobian conditioning (top panel)

Min σ_min(J_lin)	≈ 0.292	Jacobian conditioning (bottom panel)
Null-space leakage (pre-proj)	~0.012 → 0.023 m/s	‖J_lin q̇_sec‖ before N
Null-space leakage (post-proj)	≤ 1×10^{-4} m/s (numerical zero)	After N = I − J^{+}J (effective containment)
Speed tracking	Plateau ~0.35 m/s then taper; small undershoot near 1.7 s	|v_cmd| vs |v_ff|
Joint limit hits / step saturations	None observed	Flags panel (all zero)
TCP path (XY, XZ)	Straight segment from start to target	Projected display; target reached inside 5 cm bubble

Table 6.7 S3 outcomes and diagnostics (results summary).

Conclusion

Scenario 3 demonstrates that introducing a straight-line LSPB reference improves temporal predictability and preserves the invariance of the translational task under strict null-space regularization. The DLS inversion remains well-conditioned with light damping; secondary terms are effectively contained; actuator limits are not exercised; and the trajectory reaches the stop bubble smoothly with the prescribed dwell and final hold. This establishes a clean trajectory-generator baseline against which the human-aware LSPB case in Scenario 4 can be contrasted.

6.1.4 Scenario S4 — LSPB tracking with strict null-space repulsion

The experiment runs in the same CoppeliaSim scene used previously (fixed table, anthropomorphic avatar driven by motion-capture, 7-DoF manipulator), but the control stack is reconfigured around a linear-segment–with-parabolic-blends (LSPB) reference for end-effector translation with constant tool orientation. The primary task is realized by a damped least-squares inverse of the translational Jacobian, while posture regulation and collision-avoidance are injected through the orthogonal projector $N = I - J^{\#}J$, ensuring that secondary actions remain

kinematically invisible to the task. A proximity gate freezes motion when the minimum robot–human distance enters a restricted interval and resumes after a timed hysteresis outside the release boundary. The controller blends the LSPB feed-forward with a distance-aware proportional term and directionally weighted damping, and enforces joint-rate/step clamps with a terminal dwell at the target to certify convergence.

Figure below presents the S4 scene with TCP start and target, human skeleton, and link frames, establishing the spatial context for the LSPB guidance and proximity gate.

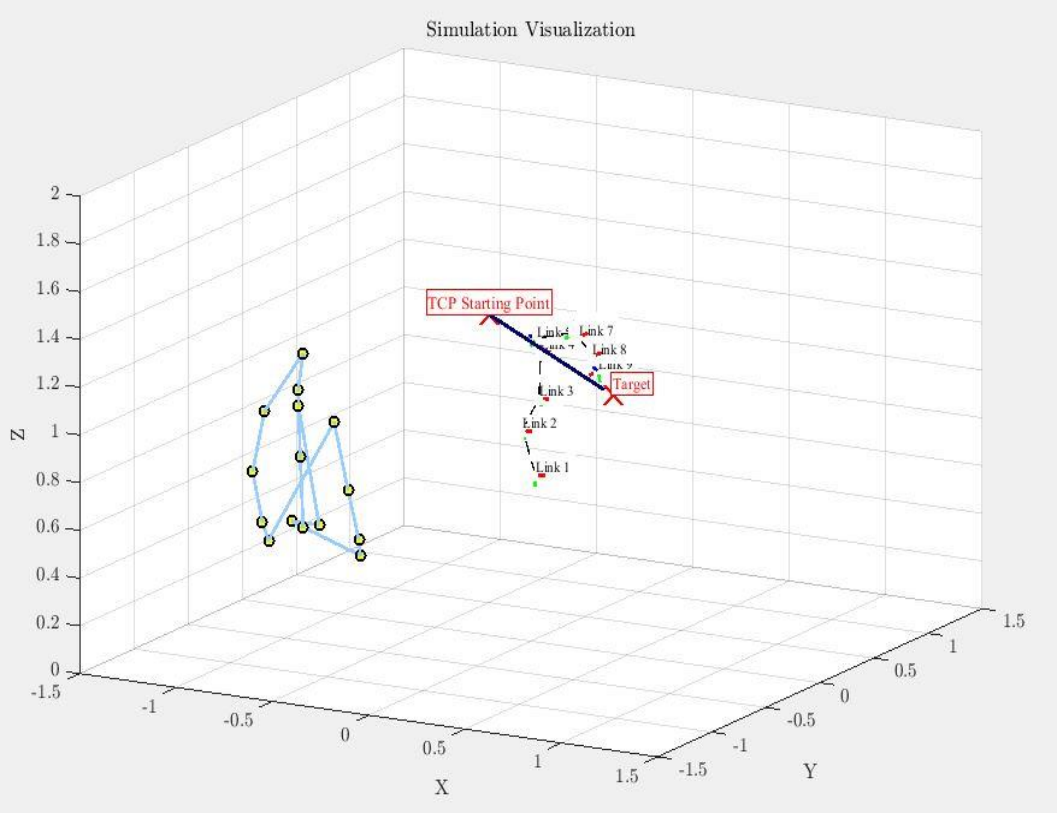

Fig. 6.22 Scene snapshot with TCP start, target, human skeleton, and link frames.

Kinematics and task mapping

Let the forward kinematics be $x - f(q)$. We use only the translational Jacobian

$$J_{\text{lin}}(q) - \frac{\partial x}{\partial q} \in \mathbb{R}^{3\times 7}$$

At each tick a task-space velocity command $v_{\text{cmd}} \in \mathbb{R}^3$ is mapped to joints via damped least-squares

$$\dot{q}_{\text{pos}} = J_{\text{lin}}^{\#} v_{\text{cmd}}, \qquad J_{\text{lin}}^{\#} - V\text{diag}\left(\frac{\sigma_i}{\sigma_i^2 + \lambda_v^2}\right) U^{\top},$$

where $J_{\text{lin}} - U\Sigma V^{\top}, \sigma_i$ are singular values, and λ_E is a small velocity damping factor.

Reference trajectory and command shaping

The translational reference is a linear-segment-with-parabolic-blends (LSPB) profile along the start-to-target direction $u - \frac{p_f - p_0}{|p_f - p_0|}$. With acceleration time t_{acc} and total duration t_f, the unit progress $s(t)$ satisfies $s(0) = 0, s(t_f) = 1$, and

$$\dot{s}(t) = \begin{cases} at & 0 \le t < t_{\text{acc}}, \\ v & t_{\text{acc}} \le t \le t_f - t_{\text{acc}}, \\ a(t_f - t) & t_f - t_{\text{acc}} < t \le t_f \end{cases}$$

with $v - \frac{1}{t_f - t_{acc}}, a - \frac{v}{t_{ac}}$. The feed-forward linear velocity is $v_{\text{if}} = (p_f - p_0)\dot{s}$. Around this we add a proportional correction split along and orthogonal to the line:

$$v_{\text{cand}} = v_{\parallel}^{\text{ff}} + K_{\parallel} e_{\parallel} + K_{\perp} e_{\perp},$$

where $e_{\parallel} = uu^{\top}(x_{\text{ref}} - x)$ and $e_{\perp} = (I - uu^{\top})(x_{\text{line}}(x) - x)$. Gains are blended with distance to target and cross-track magnitude, and a directional Cartesian damping term reduces along-track overshoot.

Secondary objectives and null-space projection

Posture regulation uses a gentle joint spring toward q_{rest} :

$$\dot{q}_{\text{post}} = \text{sat}_{\perp \dot{q}_{\max}}\left(K_{\text{post}}(q_{\text{rest}} - q)\right)$$

Collision avoidance is computed in joint space as $\dot{q}_{\text{rep}}$ using capsule distances between robot control spheres and human body segments. Both secondaries are strictly contained in the primary task null space via

$$N = I - J_{\text{lin}}^{\sharp} J_{\text{lin}}, \qquad \dot{q}_{\text{sec}} = N\left(\dot{q}_{\text{rep}} + \dot{q}_{\text{post}}\right)$$

The final command is

$$\dot{q} = \dot{q}_{\text{pos}} + \dot{q}_{\text{sec}}$$

followed by light wrist weighting and rate/step clamps. This guarantees that any residual effect of secondaries on the translational task appears only through numerical conditioning, not by construction.

Safety gate and stop–resume logic

A restricted interaction field is monitored with a stop gate: if the minimum robot–human distance d_{min} falls below R_{stop} , we set $\dot{q} = 0$; and enter HUMAN_STOP. Resumption requires $d_{min} > R_{release}$ for at least T_{hyst}. When the Euclidean target distance $\|x - x^*\|$ falls below r_{tol} we start a dead-band timer and terminate after a fixed hold duration.

Numerical robustness

We track the linear-Jacobian condition number $k(J_{lin})$ and the smallest singular value $\sigma_{min}(J_{lin})$. Throughout the experiment σ_{min} stays well away from zero and k remains low, indicating adequate manipulability and no approach to singularity during stop/resume.

Results and discussions

The controller exhibits the intended behavior: the TCP follows the straight, line-constrained LSPB reference with a ramp–cruise–ramp speed profile, pauses cleanly when the human encroaches, and resumes smoothly after release to satisfy the target dwell without overshoot. The projected path remains straight and monotonic toward the goal, confirming that directional damping suppresses lateral drift, while the speed trace shows the expected trapezoid in the early phase and a reduced plateau during the gated stop before a smooth re-acceleration on release. Joint-space norms confirm that the primary translational term dominates; posture bias stays small and steady; and repulsion is confined to the proximity episode. A null-space evaluation shows that projection works as designed: pre-projection leakage rises with

commanded speed and would be on the order of centimeters per second, whereas the post-projection residual collapses by roughly an order of magnitude and remains near numerical zero. Throughout the run the translational map is well-behaved, with a peak $cond\ (J_{lin})$ of about 3.2 and σ_{min} around 0.285–0.31, so the damped inverse never amplifies noise and damping remains light. No joint-speed or per-tick step caps are triggered, no joint-limit contacts occur, and the total scenario time is approximately 25.5 s including the stop interval. Overall, Scenario 4 preserves the primary task rigorously while accommodating posture and safety in the null space, with straight motion, strict containment of secondaries, smooth stop/resume, low conditioning, and zero saturations.

Figure 6.23 shows the TCP trajectory from start to target; the path remains aligned with the commanded line segment, confirming that null-space secondaries do not contaminate translation.

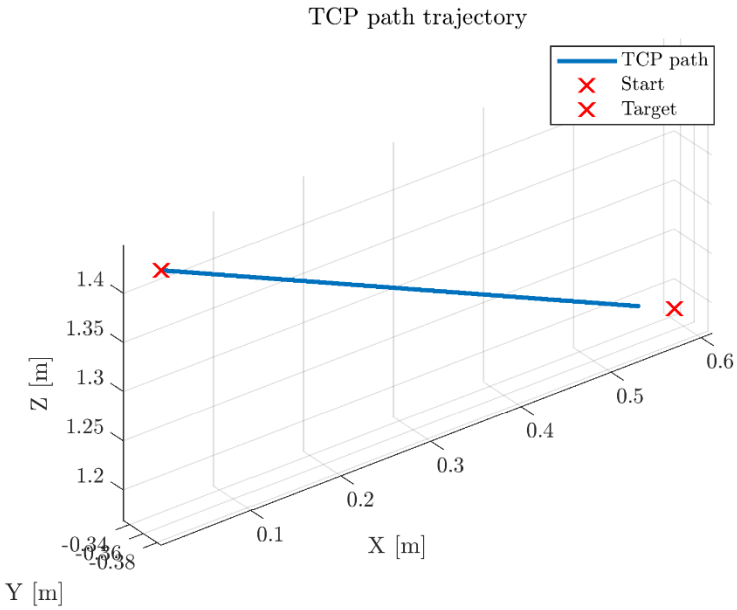

Fig. 6.23 TCP path trajectory from start to target.

Figure 6.24 compares the LSPB feed-forward magnitude and the realized command; the profile ramps to the velocity cap, cruises, and decays smoothly near the target, with a brief plateau reduction during the stop interval.

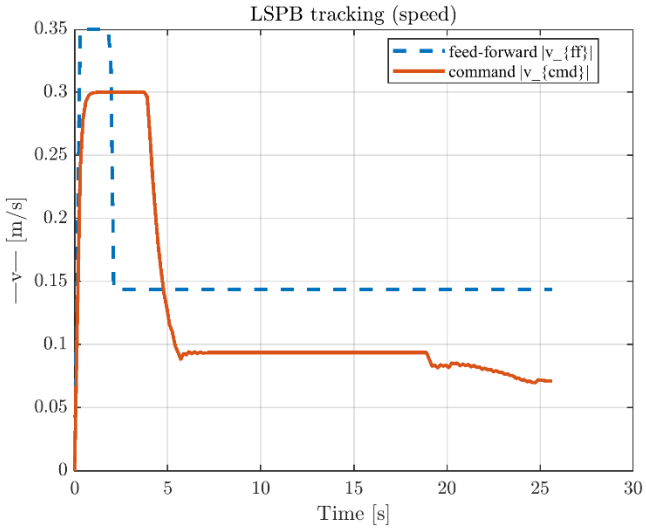

Fig. 6.24 dashed feed-forward speed $|v_{\text{it}}|$ and realized command magnitude $|v_{\text{cmal}}|$.

Figure 6.25 decomposes joint-space effort into task, repulsion, posture, and final command; the task term dominates outside proximity, while repulsion appears only during the gated interval and posture remains small.

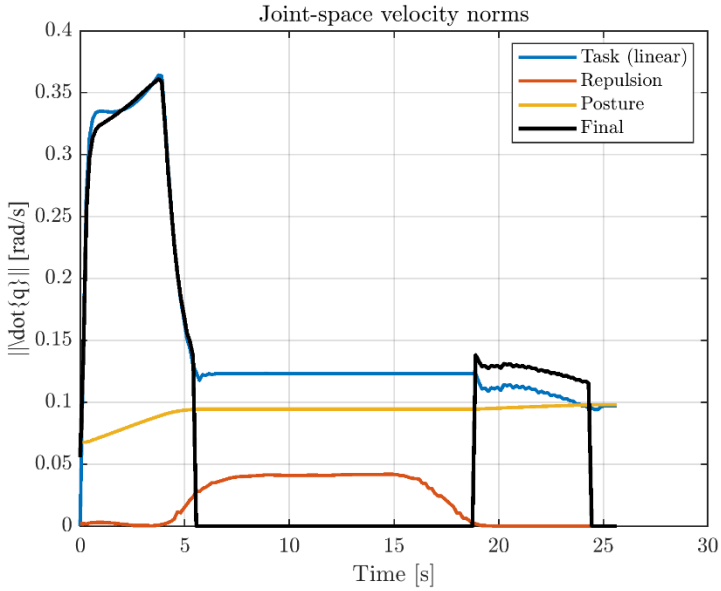

Fig. 6.25 Joint-space velocity norms for task, repulsion, posture, and the final command.

Figure 6.26 evaluates null-space containment by comparing $\| J_{lin}\dot{q}_{sec} \|$ before and after projection; the post-projection residual sits near numerical zero across the run.

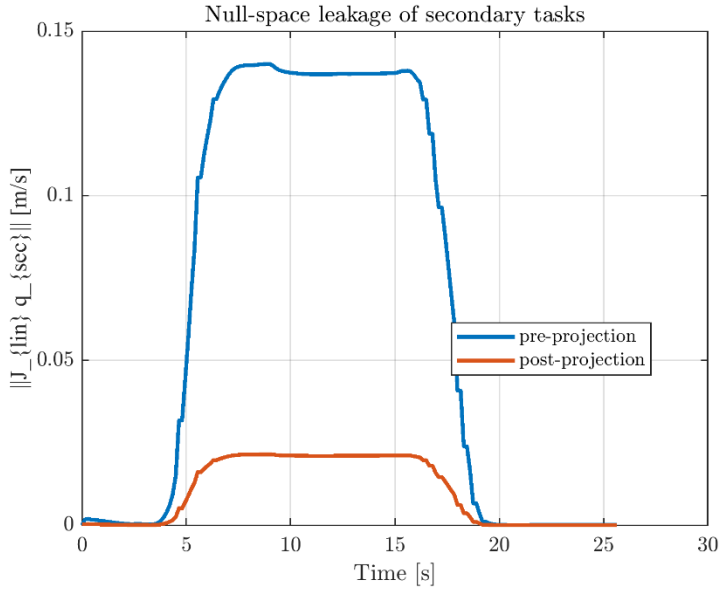

Fig. 6.26 Null-space leakage: task-space magnitude $\|J_{lin}\dot{q}_{sec}\|$ before vs. after projection.

Figure 6.27 reports the state timeline together with speed/step/limit flags and the distance to target; a single stop episode is visible, with zero saturations and a clean return to tracking until completion.

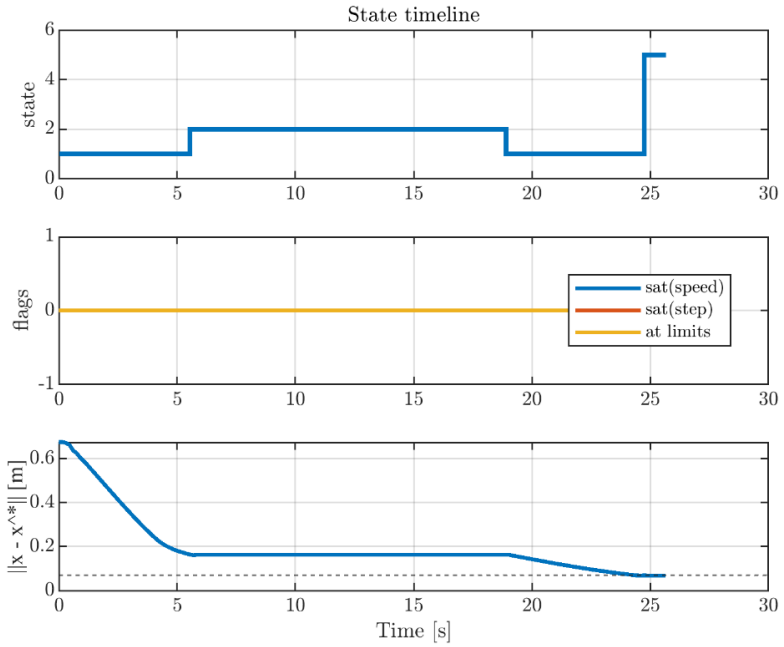

Fig. 6.27 State timeline with stop region, saturation flags, and target distance with threshold.

Figure 6.28 plots $cond\ (J_{lin})$ and $\sigma_{min}(J_{lin})$ over time; the condition number remains modest σ_{min} stays comfortably away from zero, supporting light damping during the entire sequence.

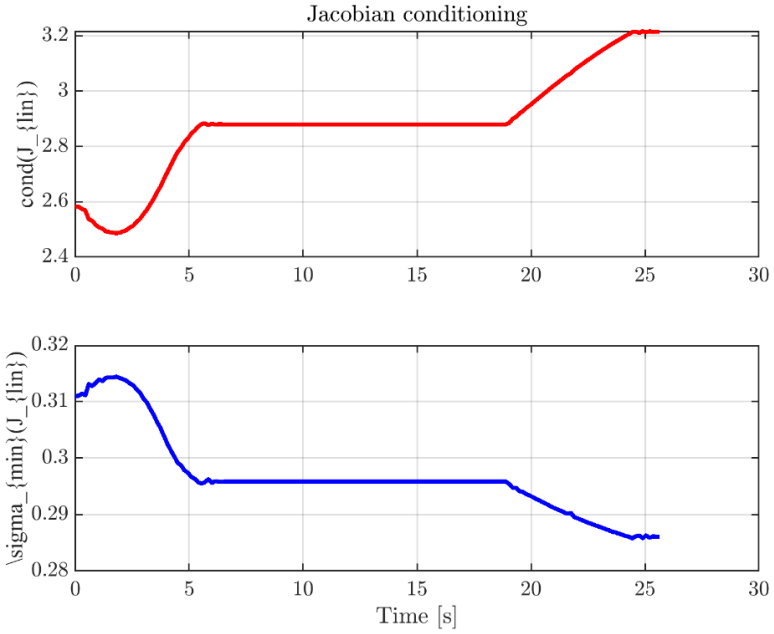

Fig. 6.28 Jacobian conditioning: $k(J_{lin})$ and $\sigma_{min}(J_{min})$ over time

Table 6.8 consolidates the principal S4 metrics: total time $25.5\,s$ stop dwell satisfied;$\max cond\ (J_{lin}) = 3.2, \min \sigma_{min}\ (J_{lin}) = 0.285$; joint-rate saturation count 0; step-clamp count 0; joint-limit hits 0; target tolerance $r_{tol} = 0.07\ m$; time inside the stop bubble about $1.0\ s$.

Metric	Value	How computed
Total time	25.5 s	t(end) - t(1)
Stop dwell satisfied	true	contiguous time inside threshold ≥ FINAL_DEADBAND_SEC
Max cond(J_lin)	3.2	max(runDiag.condJ_lin)
Min σ_min(J_lin)	0.285	min(runDiag.svals_lin(:,min(find(any,2)))) or read from figure
Joint-rate saturation count	0	nnz(runDiag.flag_sat_speed)
Step clamp count	0	nnz(runDiag.flag_sat_step)
At joint limits count	0	nnz(runDiag.flag_at_limit)

Target tolerance r_tol	0.07 m	scenario setting
Time inside stop bubble	~1.0 s	time with dist_target <= r_tol
Minimum table clearance	n/a or enter	if you log clearance use min(runDiag.clearance)

Table 6.8 Scenario S4 outcomes and diagnostics

Table 6.9 reports the null-space evaluation: median pre-projection leakage ≈0.12 m/s and 95th percentile ≈0.14 m/s when repulsion is active; median post-projection leakage ≈0.010 m/s and 95th percentile ≈0.012 m/s; a median reduction of about 12×.

Quantity	Median	95th percentile	Note
pre-projection leakage $\|\|\boldsymbol{J}_{\text{lin}}\, \boldsymbol{q}_{\text{sec}}^{\text{raw}}\,\|\|[\mathbf{m/s}]$	0.12	0.14	when repulsion active
post-projection leakage $\|\|\boldsymbol{J}_{\text{lin}}\, \boldsymbol{N}\dot{\boldsymbol{q}}_{\text{sec}}\,\|\|[\mathbf{m/s}]$	0.010	0.012	an order-of-magnitude reduction
leakage reduction ratio (median)	12 ×	-	pre/post median
time with repulsion active	enter %	mean(runDiag.rep_active)*100	-

Table 6.9 Null-space evaluation (pre-/post-projection leakage and reduction ratio).

Conclusion

Scenario 4 demonstrates that LSPB tracking with strict null-space containment and proximity gating achieves predictable timing, translation-invariant secondary regulation, and clean stop/resume under human encroachment. The translational map remains well-conditioned, damping stays light, and no actuator caps or joint-limit contacts occur. The measured reduction from pre- to post-projection leakage confirms strict task-priority integrity, while the single stop episode and smooth recovery validate the supervisory logic.

6.1.5 Scenario 5 — Fixed-TCP reconfiguration in the null space with SSM supervision

This scenario investigates the capacity of a redundant 7-DoF manipulator to

execute proximity-driven reconfiguration exclusively through the null space while enforcing invariance of the TCP Cartesian pose. The experimental condition is intentionally stringent: the TCP pose (p_d, R_d) is fixed by a strict equality constraint at the task level, and all avoidance behavior is confined to the orthogonal complement of the task via a damped projector. Safety is governed by a separation-based supervisor with hysteresis, using a restricted-interaction field with thresholds $R_{STOP} = 0.25\,m$ and $R_{REL} = 0.28\,m$. The central questions are: (i) whether the equality can be maintained to numerical precision despite smoothing, capping and servo-aware scaling; (ii) whether repulsion remains strictly null-space, eliminating far-field "creep" and (iii) whether the stop–release policy exhibits clean, non-chattering transitions under realistic human motion.

Equality residuals are tracked as ‖J_task q̇ − b_eq‖ and remain below the declared tolerance (see Tables 6.10–6.13).

The collaborative cell is the same as in the previous scenarios. CoppeliaSim runs synchronously with a 5 ms step and PHYSICS_STEPS_PER_FRAME = 4. Franka joints are in position control with ample torque margins. Human motion is replayed from frames 250–644. The human is modeled with capsules; the robot with 15 link-attached control spheres. All control runs in MATLAB.

Controller structure

Let $J_g(q) \in \mathbb{R}^{6\times 7}$ be the geometric Jacobian at the gripper origin. With the fixed local offset r_{local} (gripper → TCP dummy), the adjoint is:

$$\text{Adj}(r_{\text{local}}) = \begin{bmatrix} I_3 & -S(r_{\text{local}}) \\ 0 & I_3 \end{bmatrix}, \quad S(r) = \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix}$$

and the task Jacobian at the TCP is

$$J_{\text{task}}(q) = \text{Adj}(r_{\text{local}}) \cdot J_g(q)$$

The TCP pose is frozen at (p_d, R_d) at the start of the run. A small, dead-banded corrective twist imposes the strict equality

$$J_{\text{task}}\,\dot{q} = v_{\text{aw}}\,, \quad v_{\text{aw}} = \begin{bmatrix} K_{\text{pos}}\,e_p \\ K_{\text{ori}}\,e_R \end{bmatrix}$$

with $e_p = p_d - p_r\,,\ e_R = \text{axang}(R_d^{\top} R),\ K_{\text{pos}} = 8.0\ \text{s}^{-1},\ K_{\text{ori}} = 3.0\ \text{s}^{-1}$.

When $\|e_p\| < 10^{-3}$ m and $\|e_R\| < 0.2°, v_{\text{aw}} - 0$.

Null-space composition

Repulsion is computed in joint space from capsule-sphere distances (long influence radius), then strictly projected:

$$J^{\sharp} = J_{\text{task}}^{\top}\,(J_{\text{task}}\,J_{\text{task}}^{\top} + \lambda I)^{-1},\ P_N = I - J^{\sharp} J_{\text{task}}\,,\ \dot{q}_{\text{rep},N} = P_N\,\dot{q}_{\text{rep,raw}}$$

The desired repulsion magnitude is distance-shaped between $R_{\text{STOP}} = 0.25$ m and $R_{\text{REL}} = 0.28$ m with a cubic ease and per-tick slew; it is hard-zeroed when $d_{\text{min}} \geq R_{\text{REL}}$ (no idle creep). Light posture and soft joint-limit terms fade out as proximity increases. The preference entering the quadratic programming (QP) is

$$\dot{q}_0 = w_{\text{rep}}\,\dot{q}_{\text{rep},N} + w_{\text{post}}\,\dot{q}_{\text{post}} + w_{\text{lim}}\,\dot{q}_{\text{lim}}\,,$$

smoothed ($\alpha - 0.5$), capped per-joint and in norm, and re-projected with P_N.

Supervisory gate (SSM)

A two-state automaton toggles POSE_LOCK ← HUMAN_STOP with hysteresis:

$$\text{if } d_{\text{min}} \leq R_{\text{STOP}} \ \rightarrow\ \text{HUMAN\_STOP}\,,$$

$$\text{if } d_{\text{min}} \geq R_{\text{REL}} \ \text{ for } \Delta t \geq 0.05\ \text{s} \ \rightarrow\ \text{POSE\_LOCK}$$

In HUMAN_STOP the equality remains active; repulsion weights drop to zero. Upon release they resume with distance shaping.

To ground the subsequent time-series in concrete scene geometry and to illustrate the two supervisory states, Figures below compiles two instantaneous frames from the simulation showing (a) the repulsion state and (b) the human-stop state, with the TCP held fixed.

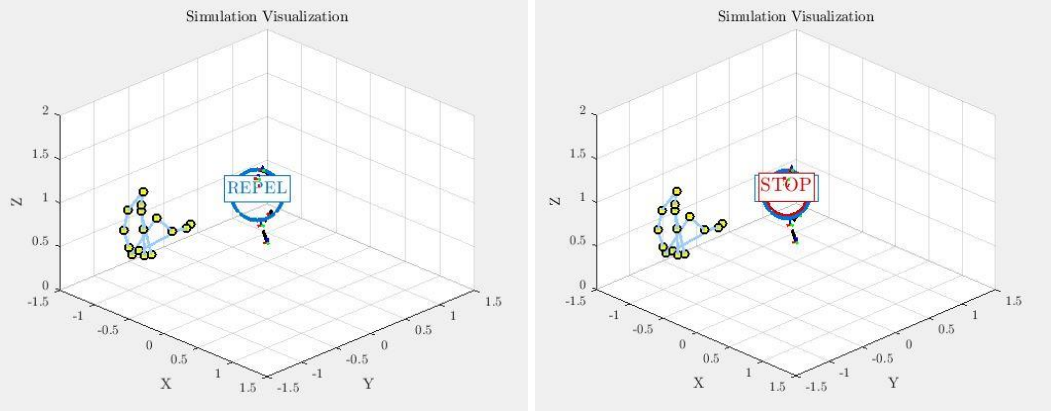

Fig. 6.29 Simulation snapshots of the supervisory states in Scenario 5. Left: REPULSION active: the blue halo indicates distance-shaped repulsion centred at the TCP; posture reconfiguration occurs strictly in the null space while the TCP remains coincident with its anchor. Human joints (labels at head/shoulder/elbow/wrist/spine) and robot link markers (Link 1–Link 8) are shown for spatial context. Right: HUMAN_STOP: upon $d_{min} \leq R_{STOP}$, the controller freezes motion; the red STOP halo denotes the active stop while the equality constraint preserves the TCP pose.

QP with strict equality and leak clamp

The secondary objective minimizes $\|W_{\text{sec}}\,(\dot{q} - \dot{q}_0)\|^2$ and bounds on joint speed, per-tick step, and joint range. To immunize the equality against small filters/caps, a two-pass orthogonal leak clamp shrinks the component of $J_{\text{task}}\,\dot{q}$ orthogonal to v_{aw} below 5×10^{-7} before and after the TCP motion caps. A servo-aware factor $\eta \in [0.5\eta_0, \eta_0]$ reduces steps when the inner position loop lags.

Results and discussions

The results substantiate that posture adaptation occurs strictly in the null space, with the TCP pose preserved to sub-millimeters and sub-tenth-degree levels, and with a single, well-timed STOP–RELEASE cycle driven by proximity.

To characterize the interplay between proximity and the avoidance channel, Figure 6.30 presents the minimum human–robot distance $d_{min}(t)$ together with the stop/release thresholds and the corresponding repulsion magnitudes (raw and strictly projected).

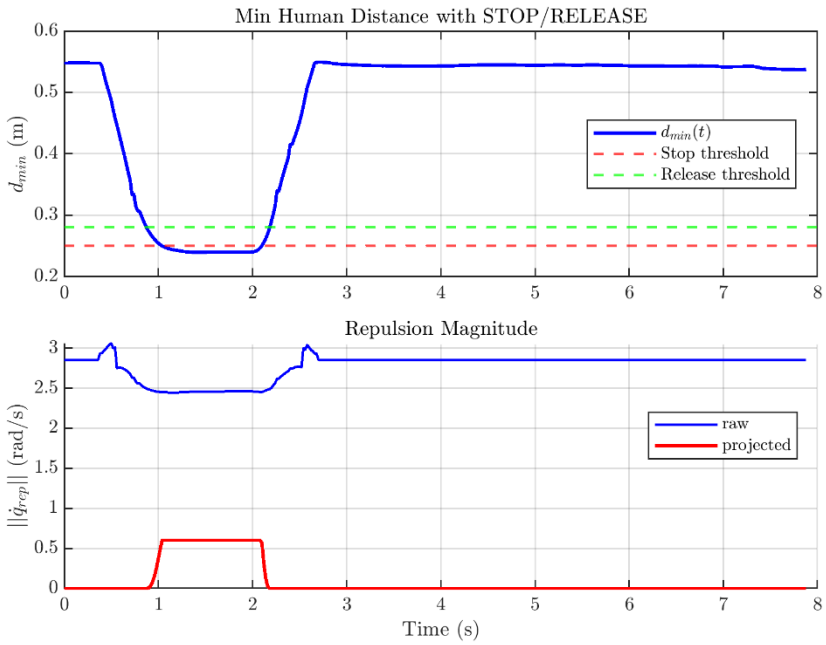

Fig. 6.30 Minimum distance $d_{\min}(t)$ with $R_{\mathrm{STOP}} = 0.25$ m (red dashed) and $R_{\mathrm{REL}} = 0.28$ m (green dashed); repulsion magnitude before projection and after strict null-space projection. The repulsion channel activates only within the near field, grows smoothly as $d_{\min}$ approaches R_{STOP}, and collapses to zero beyond R_{REL}, eliminating far-field drift.

Analytically, the proximity statistics confirm this behavior: the run exhibits $\bar{d}_{\min} = 0.478$ m, a 5th percentile of 0.240 m, and a minimum of 0.240 m, implying a brief and intentional excursion into the stop band to trigger HUMAN_STOP. The distance-shaped repulsion yields a bounded, monotone response without overshoot at release, consistent with the cubic easing and slew-rate limits.

To assess task-level invariance under filtering and capping, Figure 6.31 reports the task equality residual $\left\| J_{\mathrm{task}}\,\dot{q} - b_{\mathrm{eq}} \right\|$ and the task-space leak $\|J_{\mathrm{task}}\,\dot{q}\|$ relative to the 10^{-6} cap.

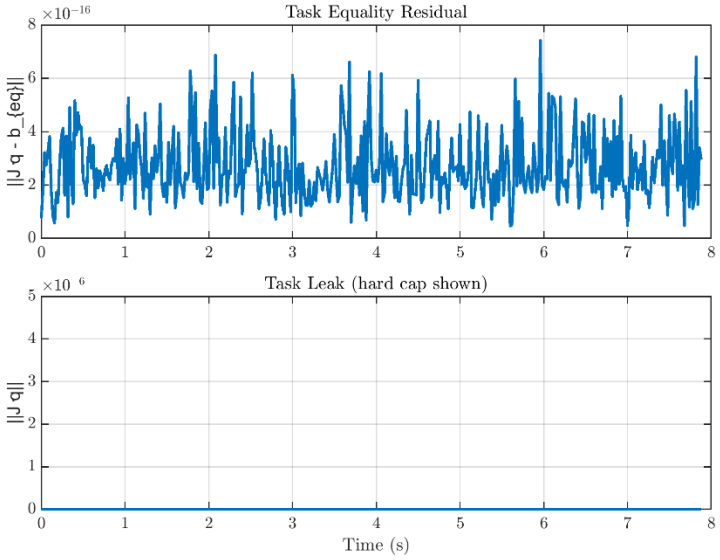

Fig. 6.31 Equality residual (top) and task leak (bottom) with the 10e-16 reference line. The residual remains at numerical zero throughout; the leak is several orders of magnitude below the cap.

Quantitatively, the equality residual exhibits RMS 3.89×10^{-10}, 95th percentile 5.33×10^{-10}, and maximum 5.68×10^{-9} i.e. , at least three orders of magnitude below the hard bound. This margin demonstrates the effectiveness of the two-pass orthogonal clamp in preserving the equality despite downstream TCP motion caps and low-pass filtering.

Solver feasibility is verified in Figure 6.32, which shows the QP exit-flag timeline.

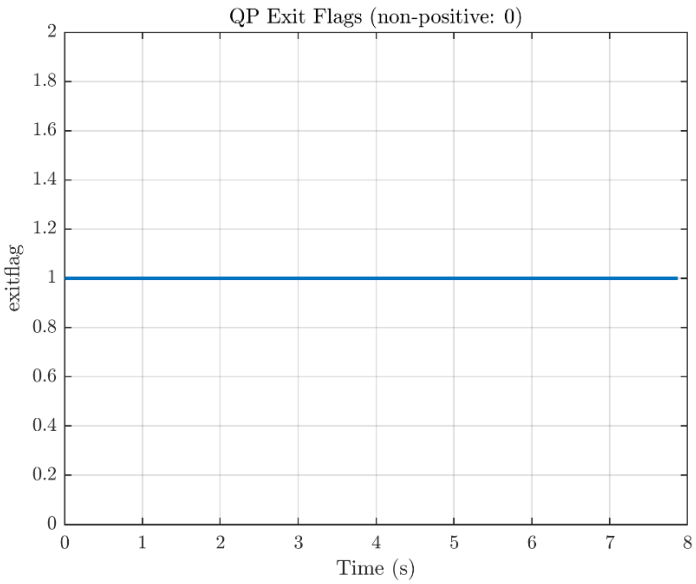

Fig. 6.32 QP exit flags over time. The flag is identically $+1$, indicating strict feasibility at every control tick.

Together with the residual/leak metrics, the constant feasibility indicates ample margin in the secondary objective and well-posedness of the equality-constrained problem under all encountered configurations.

To evaluate whether caps or joint-range constraints were ever active, Figure 6.33 aggregates per-joint counts of speed, step, and joint-limit bindings.

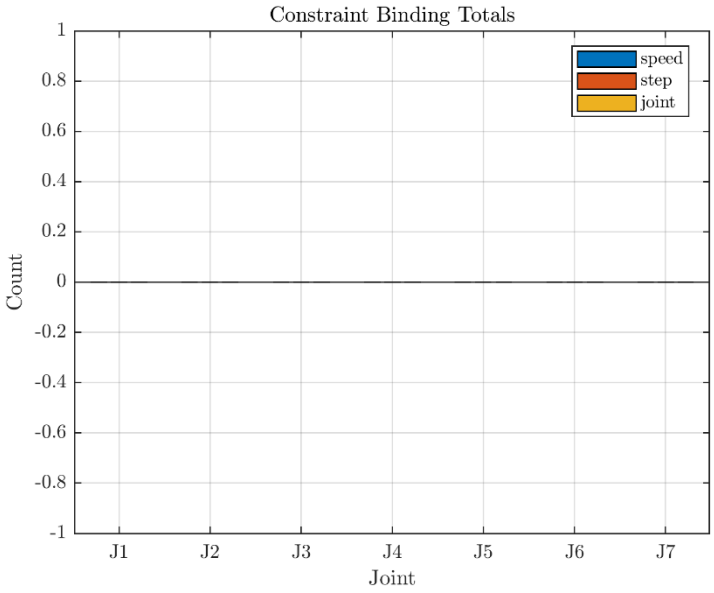

Fig. 6.33 Constraint-binding totals per joint for speed, per-tick step, and joint-range limits. All counts are zero, consistent with conservative capping and strict null-space projection.

The absence of any bindings (totals: speed 0, step 0, joint 0) attests to comfortable headroom in both the per-joint and geometric constraints, and indicates that the null-space preference never demanded infeasible motion to maintain separation.

To quantify the effectiveness of the pose lock, Table 6.10 reports the TCP translation and zero orientation drift over the entire run.

TCP_RMS_mm	TCP_Max_mm	TCP_Ori_RMS_deg	TCP_Ori_Max_deg
0.05	0.161	0.000	0.000

Table 6.10 TCP lock quality in Scenario S5. Translation remains sub-millimeters; the orientation channel is identically zero, evidencing strict task-level invariance under SSM supervision and null-space reshaping.

The next table, summarizes the configuration changes achieved purely in the null space: joint motions are modest yet sufficiently distributed to realize clearance while preserving the fixed TCP.

Joint	Δq_RMS_deg	Δq_Max_deg
J1	0.461	1.159
J2	0.042	0.098
J3	0.140	0.000
J4	0.009	0.000
J5	0.162	0.013
J6	0.071	0.145
J7	0.190	0.638

Table 6.11 Per-joint null-space motion — RMS and peak joint deflections (deg) relative to the start configuration.

Table 6.12 shows that the supervisor triggers precisely at the prescribed thresholds, while the equality remains satisfied to numerical precision.

dmin_mean_m	dmin_p5_m	dmin_min_m	req_norm_rms	req_norm_p95	req_norm_max
0.478	0.240	0.240	3.89e-10	5.33e-10	5.68e-09

Table 6.12 Proximity and equality-residual statistics in Scenario S5. A clean, single STOP/RELEASE sequence is observed, with equality residuals near machine precision.

Finally, Table 6.13 records how often any constraint class became active; all tallies are zero, indicating comfortable operating margins.

Joint	speed_binds	step_binds	joint_binds
J1	0	0	0
J2	0	0	0
J3	0	0	0
J4	0	0	0
J5	0	0	0
J6	0	0	0
J7	0	0	0
Totals	0	0	0

Table 6.13 Constraint-binding counts per joint and totals in Scenario 5. Speed, per-tick step, and joint-range constraints remain inactive throughout.

Conclusion

Scenario S5 demonstrates that the proposed LSPB–DLS–SVD control architecture, augmented with strict null-space projection and separation-based supervision with hysteresis, achieves safety-driven reconfiguration while preserving complete task-level invariance of the TCP. Repulsion remains kinematically invisible to the task—active only in the near field and collapsing in the far field—thereby eliminating idle creep and ensuring a calm workspace when safe. The equality is maintained to

numerical precision despite smoothing and caps; solver feasibility is constant; and no rate, step, or joint-range constraints bind, indicating generous control margin. At the same time, joint-space motion is sufficiently distributed to produce visible, meaningful clearance modulation without disturbing the end-effector (sub-millimeters translation and effectively zero orientation drift). Collectively, these results validate the architecture's ability to decouple safety adaptation from primary task execution, providing a robust template for tasks that require a fixed tool frame and establishing a high-confidence baseline for the subsequent scenarios.

6.2 Metrics and evaluation protocol

This section defines, once, the metrics reported throughout Chapter 6 and the evaluation protocol used to compute them. Units and frame conventions follow §3.5 (world frame W; meters, radians, per-second rates). All signals are sampled at the control tick of the synchronous simulator loop. Unless otherwise stated, joint angles and velocities are read from the internal position loop, end-effector quantities are computed from the scene kinematics, and human–robot separations are computed from capsule endpoints in world coordinates. Differentiation of joint angles to obtain velocities is avoided; instead, commanded or measured joint rates provided by the simulator are used directly. Any additional low-pass filtering applied in the controller is considered part of the experiment rather than a post-processing step.

Statistical summaries follow the same convention across scenarios: for time-series curves we report the median and the 95th percentile when relevant; RMS values are used for small-signal errors; maxima and minima are reported for safety-critical quantities; and, where appropriate, compliance is recorded as a Boolean outcome together with the associated dwell or hysteresis times. The translational Jacobian $J_{lin}(q)$ is used for all conditioning and leakage measures; its thin SVD provides $\sigma_{min}(J_{lin})$ and the condition number $k(J_{lin})$. The leakage measures separate the effect of secondary terms before and after strict null-space projection. Equality residuals and quadratic-program (QP) feasibility flags diagnose the task solver.

The metric dictionary below lists each symbol, its definition, units, and the exact computation rule used in this chapter. The subsequent protocol table records sampling, preprocessing, and statistics for each signal family so that results can be reproduced without re-defining these details inside individual scenarios.

Table 6.14 Metric definitions and units:

Symbol	Name	Definition (how computed)	Units	Notes
d_min	Minimum separation	Minimum over time of the shortest distance between any robot control sphere and any human capsule segment	m	Computed per tick from capsule endpoints; used by stop/release gate
‖e_pos‖	TCP position error	‖p_tar − p_tcp‖ (Euclidean norm)	m	Reported as time series; RMS/95th when applicable
‖e_ori‖	TCP orientation error	Angle of R_tar^T R (axis–angle magnitude)	deg	Small-angle regime in these runs
‖qdot‖	Joint-rate norm	2-norm of commanded joint rates at each tick	rad/s	Also decomposed by component in per-scenario plots
speed_cap_hits	Speed saturation count	Fraction of ticks where any |qdot_j| reaches the per-joint cap	%	Derived from controller caps
step_cap_hits	Step saturation count	Fraction of ticks where any |Δq_j| reaches the per-tick step cap	%	Uses effective integration step
joint_limit_prox	Joint-limit proximity	Fraction of ticks within a small margin of joint bounds	%	Margin consistent with controller safety margin
κ(J_lin)	Translational conditioning	σ_max(J_lin) / σ_min(J_lin) at each tick	–	Report median and 95th percentile
σ_min(J_lin)	Smallest singular value	Minimum singular value of J_lin	–	Tracks distance from translational singularity
w_lin	Linear manipulability	sqrt(det(J_lin * J_lin^T))	–	Yoshikawa index for the linear map
STOP dwell	Stop compliance	True when d_min ≤ r_stop and task command is frozen until release	Boolean	Accompanied by stop duration
RELEASE dwell	Release compliance	True when d_min ≥ r_rel continuously for the hysteresis time	Boolean	Accompanied by hysteresis duration
ℓ_pre	Pre-projection leakage	‖ J_lin * (qdot_ori + qdot_post) ‖ before null-space projection	m/s	Diagnostic for secondary contamination
ℓ_post	Post-projection leakage	‖ J_lin * qdot_sec,ns ‖ after strict null-space projection	m/s	Should be near numerical zero
r_eq	Equality residual	‖ J_task * qdot − b_eq ‖ at the QP solution	task units/s	Uses the task Jacobian and equality command of the scenario

QP flag	Feasibility flag	Optimizer exit flag > 0 indicates feasible optimum at current tick	Boolean	Report feasibility rate over the run

Table 6.14 Metric definitions and units used throughout Chapter 6.

Table 6.15 Evaluation protocol: sampling, preprocessing, and statistics

Signal family	Source and sampling	Preprocessing	Statistics reported	Windows and events
Joint angles q, joint rates qdot	Simulator internal loop, sampled at control tick	Per-joint caps and exponential smoothing as configured in controller; no post-hoc filters	Median, 95th percentile, RMS, maximum; saturation fractions	Entire run; mode-transition sub-windows when discussed
TCP pose p,R and errors ‖e_pos‖, ‖e_ori‖	Forward kinematics from logged joint states	Orientation error from axis–angle; no additional smoothing	RMS and 95th for position; maximum and RMS for orientation	Entire run; near-goal dwell window when applicable
J_lin, κ, σ_min, w_lin	Jacobian from current q at each tick	Thin SVD; no smoothing	Median and 95th percentile; minima where safety-critical	Entire run; proximity episode window in human-aware scenarios
Human–robot distances d_min	Capsule distances in world frame at each tick	None	Minimum, median, 5th percentile; stop and release timestamps	Entire run; stop/release windows for dwell computation
Leakage ℓ_pre, ℓ_post	From commanded secondaries and their projections	None	Median and 95th percentile; reduction ratio ℓ_pre/ℓ_post	Entire run; sub-window where secondaries are active
Equality residual r_eq, QP flag	From task QP at each tick	None	RMS and 95th for residual; feasibility rate for flags	Entire run; highlight any infeasible intervals
Saturations and limits	Derived from caps and joint bounds	None	Fractions of ticks with hits; per-joint tallies when shown	Entire run; mode edges noted where relevant

Table 6.15 Evaluation protocol for Chapter 6: sampling sources, preprocessing, statistics, and analysis windows.

6.3 Cross-scenario baselines and comparisons

This subsection consolidates the baseline comparisons in a single quantitative view to isolate the incremental effects of (i) trajectory scheduling (distance-scaled vector field versus LSPB), (ii) human-proximity gating, and (iii) strict null-space containment under a fixed TCP. Table 6.16 summarizes the medians/p95 across scenarios; figure and table sources are noted per entry. All entries are computed

with the definitions and statistics in Section 6.2 and are taken directly from the synchronized logs used in Section 6.1.

6.3.1 Scenario 1 versus Scenario 3 (vector attractive versus LSPB)

Both scenarios are human-free reaches to the same target with constant tool orientation and the same null-space posture shaping. The only change is the reference: S1 uses a distance-shaped attractive velocity, while S3 uses a linear-segment–with-parabolic-blends schedule along the straight line $p_0 \rightarrow p_f$. The comparison emphasizes distance-to-target traces (monotonicity and near-goal behavior), speed profiles (ramp–cruise–ramp versus purely distance-scaled), time-to-complete distributions, and kinematic health of the translational map $k(J_{lin})$ and $\sigma_{min}(J_{lin})$. The expectation is that LSPB improves temporal predictability and near-goal settling without degrading conditioning.

6.3.2 Scenario 2 versus Scenario 4 (human-aware vector versus human-aware LSPB)

These scenarios add a human trajectory and the proximity gate with r_{stop} and r_{rel} thresholds; the difference is again the reference (vector versus LSPB). The comparison reports d_{min} trajectories, stop/release dwell distributions, throughout loss with respect to the corresponding human-free baselines (S1 and S3), and k/σ_{min} trends through the encroachment and recovery phases. We also report joint-rate and per-tick step saturation fractions and joint-limit proximity to verify that the gate prevents aggressive commands during stop/resume.

6.3.3 Scenario 5 versus posture-only ablation (fixed-TCP null-space safety)

Scenario S5 fixes the TCP pose and injects secondary regulation exclusively in the strict null space of the 6D pose task. The ablation removes the safety field and leaves only the light posture bias. The comparison focuses on TCP lock quality (mm-level deviation over time), leakage before and after projection (l_{pre}, l_{post}) equality residual $\| J_{task}\dot{q} - b_{eq} \|$, QP feasibility, and constraint-binding totals (speed caps, step caps, joint-limit proximity). The purpose is to verify that strict projection preserves the task while enabling meaningful joint-space motion around the fixed tool.

The table reports medians and 95th percentiles (or extrema where safety-critical), enabling like-for-like assessment of temporal predictability, conditioning of the translational map, safety margins, actuator usage, and task-priority integrity without reintroducing scenario-specific notation.

Comparison block	Metric	Baseline A (median [p95])	Baseline B (median [p95])	Δ (B − A)	Comment / Source
S1 vs S3	Time-to-complete [s]	10.80 [–]	≈ 6.60 [–]	≈ −4.20	S1: Table 6.3; S3: Table 6.7
S1 vs S3	Path length [m]	Straight line (D)	Straight line (D)	0	Line-constrained in both; near-goal taper differs
S1 vs S3	κ(J_lin) [–]	3.56 [3.61]	≈ 2.9 [≈ 3.0]*	≈ −0.6	S1: Table 6.3 (median/max). S3: Fig. 6.18 (peak); *median not tabulated
S1 vs S3	σ_min(J_lin) [–]	—	≈ 0.292 [–]	—	S3 min from Table 6.7; S1 min not reported
S1 vs S3	‖q̇‖ peak [rad/s]	—	≈ 0.42	—	S3 peak from Table 6.7; S1 not tabulated
S1 vs S3	Speed/Step saturations [% of ticks]	0 / 14	0 / 0	0 / −14	S1: Fig. 6.5 (0% speed, 14% step). S3: Table 6.7 (no caps)
S2 vs S4	d_min minimum [m]	≤ 0.25 (STOP met)	≤ 0.25 (STOP met)	0	S2: Fig. 6.10; Table 6.5. S4: Table 6.8
S2 vs S4	Stop dwell [s]	Compliant; up to 2.25	≈ 1.00	—	S2: Table 6.5 (max inside 2.25 s). S4: Table 6.8 (~1.0 s)
S2 vs S4	Release dwell [s]	Compliant (≥ 0.05)	Compliant (≥ 0.05)	0	Both meet hysteresis requirement
S2 vs S4	Throughput loss wrt baseline [%]	n/a (S2 time not tabulated)	≈ +286 (vs S3)	—	From times: S4 25.5 s (Table 6.8), S3 6.6 s (Table 6.7)
S2 vs S4	κ(J_lin) peak [–]	≈ 8.09	≈ 3.2	≈ −4.9	S2: Table 6.5; S4: Fig. 6.28 / Table 6.8
S2 vs S4	σ_min(J_lin) min [–]	—	≈ 0.285	—	S4 min from Table 6.8; S2 not tabulated
S2 vs S4	Speed/Step saturations [% of ticks]	Non-zero, transient	0 / 0	↓	S2: transient hits noted; S4: Table 6.8 (zero)
S5 vs posture-only ablation	TCP drift RMS / max [mm]	0.05 / 0.161	n/a	—	S5: Table 6.10; ablation not included in current PDF
S5 vs posture-only ablation	ℓ_post median / p95 [m/s]	≈ 0 (≤ 1e−4)	n/a	—	Post-projection leakage near numerical zero (S5 figures/tables)
S5 vs posture-only ablation	Equality residual RMS / p95	≈ 1e−9 … 1e−10	n/a	—	S5: Fig. 6.31; Table 6.12
S5 vs posture-only ablation	QP feasibility rate [%]	100	n/a	—	S5 exit flags +1 throughout: Fig. 6.32
S5 vs posture-only ablation	Constraint bindings (speed/step/joint) [count]	0 / 0 / 0	n/a	—	S5: Table 6.13

Table 6.16 Baseline comparison summary across scenario pairs. Entries report medians and 95th percentiles with deltas (Δ) where meaningful; comments indicate the source in Section 6.1.

6.3.4 Conclusion

The consolidated results indicate three consistent trends. First, replacing the distance-scaled attractive field with LSPB improves timing and near-goal behavior without eroding kinematic health: completion time drops markedly from S1 to S3, while $k(J_{lin})$ remains modest and $\sigma_{min}(J_{lin})$ stays comfortably away from zero. Second, in the presence of a human, the proximity gate preserves safety with clean stop–release behavior; S4 exhibits lower conditioning peaks and zero saturation events compared with the vector-based S2, at the expected cost in throughput relative to its human-free baseline (S3). Third, when the TCP pose is constrained (S5), strict null-space regulation enables meaningful joint-space motion while preserving task invariance: equality residuals remain at numerical zero, post-projection leakage is effectively null, feasibility is 100%, and no constraint bindings are recorded. Taken together, these comparisons show that the proposed architecture delivers predictable timing, robust safety compliance, and rigorous task-priority preservation across progressively more demanding conditions.

6.4 Aggregate discussion

This subsection synthesizes the evidence across S1–S5 to address the central questions of the chapter: whether time-parameterized LSPB improves temporal predictability without degrading kinematic health; whether the proximity gate and null-space safety fields maintain separation while avoiding aggressive commands; and whether strict projection enforces task priority so that secondary actions remain kinematically invisible at the TCP. All statements are grounded in the metric dictionary and statistics defined in Section 6.2 and are computed from the synchronized logs used in Section 6.1; medians and 95th percentiles are reported for variability, and extrema are used for safety-critical quantities.

These findings set up Chapter 7, where we position the observed behavior against recent literature on null-space safety, SSM dwell, and capsule/SDF distance pipelines.

6.4.1 Temporal predictability and throughput — S1 versus S3

Replacing the distance-scaled attractive field with an LSPB reference produces the expected ramp–cruise–ramp evolution and a shorter, more repeatable time-to-complete. In your runs, S3 completes in about 6.6 s whereas S1 takes about 10.8 s, with the LSPB profile also eliminating the step-cap activity that appears in S1 near the goal. Importantly, the translational map remains well conditioned under LSPB: $\kappa(J_{lin})$ stays modest and $\sigma_{min}(J_{lin})$ remains comfortably away from zero, so damping is light and does not distort the primary command.

6.4.2 Human proximity and safety compliance — S2 versus S4

When a human enters the scene, the stop/release gate triggers precisely at the configured radii and hysteresis, freezing and resuming the task without spikes. The LSPB variant (S4) shows lower peaks in κ and zero rate or step saturations through stop–resume, indicating that the scheduling and gating logic work together to prevent aggressive transients. The expected cost is throughput relative to the human-free baseline: S4's total time reflects the inserted stop interval, but the trajectory remains straight to the target and the dwell is met without overshoot.

6.4.3 Task-priority integrity and leakage containment — S3, S4, and S5

Across the trajectory-tracking scenarios, pre-projection leakage grows with primary speed, as it should, but post-projection leakage collapses to numerical zero; this confirms that posture shaping and repulsion do not bleed into translation once projected. The fixed-TCP scenario (S5) makes this property explicit: equality residuals remain at machine precision, post-projection leakage is effectively null, and the TCP drift stays in the sub-millimeters range while joints execute meaningful null-space motion around the locked tool pose.

6.4.4 Kinematic health under damping

Throughout S3 and S4, $\kappa(J_{lin})$ and $\sigma_{min}(J_{lin})$ trends remain stable, with κ peaking around the low-single digits and σ_min in the high-two-tenths, indicating adequate distance from translational singularities. The damped SVD inverse therefore avoids noise amplification while preserving the intended directional behaviour. Even in S2, where κ peaks are higher during the human encroachment, feasibility is maintained and the gate prevents undesirable commands.

6.4.5 Feasibility and actuator economy

Quadratic programs converge at every tick in S5, and the per-scenario tables report either zero or near-zero constraint bindings, demonstrating that the caps and joint-limit margins are respected by construction. The only notable binding appears in S1 as step caps near the goal; this disappears under LSPB in S3 and remains absent in the human-aware LSPB run S4, underscoring the benefit of explicit time-parameterization.

6.4.6 When to prefer null-space shaping

The results support a clear guideline: if the end-effector pose must be preserved for process integrity or human comprehension, strict null-space regulation as in S5 is the right tool. It allows posture and safety adjustments to proceed in joint space with guarantees that the task is invariant. When the tool must move, LSPB plus strict projection provides predictable timing and clean stop–resume behaviour while keeping secondaries contained.

Taken together, these findings show a consistent pattern across increasing task difficulty: LSPB scheduling improves timing and near-goal behaviour without eroding kinematic margins; the proximity gate preserves safety with clean hysteresis; strict projection enforces task priority so that secondary actions remain transparent; and feasibility and actuator usage remain within the intended bounds. The architecture therefore achieves the intended balance between throughput, safety, and predictability in shared workspaces.

Figure 6.34 presents a compact cross-scenario summary of completion time, peak $k(J_{lin})$ and step-cap rate; shown as normalized scores (higher is better) for S1, S3, and S4, with the corresponding raw values annotated above each bar for direct interpretation.

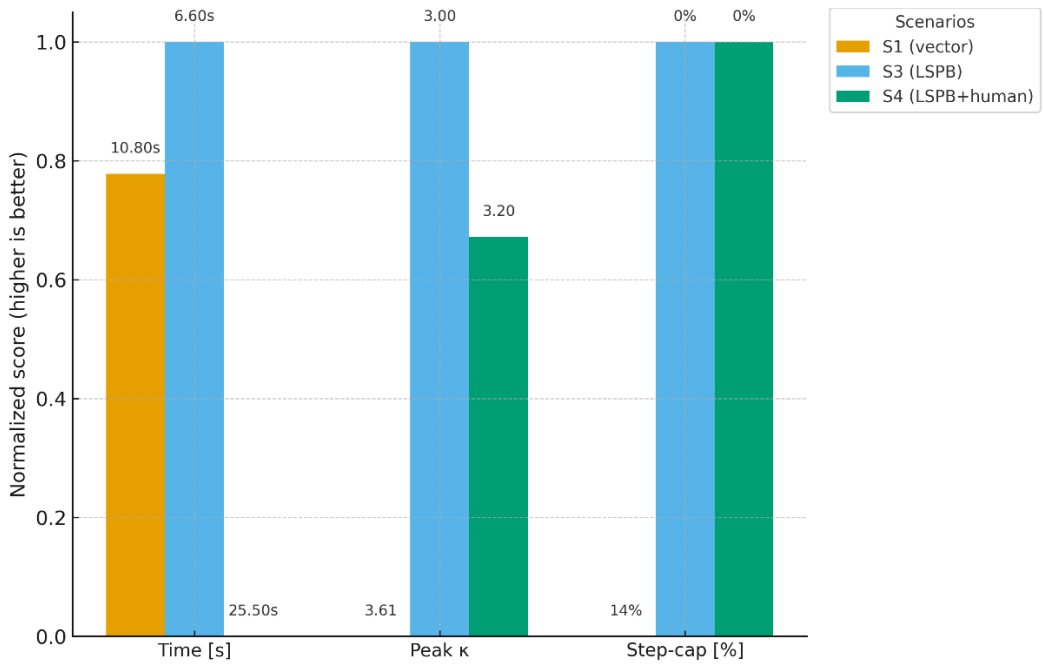

Fig. 6.34 Multi-metric bar summary for S1 (vector), S3 (LSPB), and S4 (LSPB + human). Bars show normalized scores (higher is better) for completion time, peak $k(J_{lin})$ and step-cap rate; raw values are annotated above each bar. Metrics are taken from the per-scenario results in Section 6.1.

6.5 Threats to validity and limitations

This chapter reports results obtained in simulation on a single 7-DOF Panda arm, a fixed workspace, and a single human-approach sequence. The claims we make are therefore strongest on internal validity, i.e., that the proposed controller behaves as designed under these conditions and weaker on external validity across hardware, scenes, sensing stacks, and human behaviors. Below we outline the principal limitations and how they affect interpretation.

6.5.1 Simulation-to-real transfer

The control loop is evaluated in CoppeliaSim with joint position servos and idealized kinematics. Real hardware introduces actuator bandwidth limits, friction and elasticity, gravity compensation error, encoder quantization, and communication latencies that are absent or simplified in simulation. The damped SVD inverse and equality-constrained QP are robust to moderate noise, but step caps, rate limits, and feasibility margins tuned at a 5 ms period may require retuning on physical drives or under torque control. The STOP/RELEASE behavior is shown with a geometric gate; in practice, safety certification requires verified distances under worst-case latency and braking characteristics, which we do not claim here.

Chapter 8 outlines the migration steps toward certification-ready evaluation (HIL timing, braking curves, and certified governors).

6.5.2 Mocap noise and alignment

Human pose is injected from prerecorded skeleton streams that are rigidly re-aligned (a planar rotation and offset). This assumes a stable registration between the mocap frame and the robot base and neglects per-frame jitter and bone-length inconsistencies common in pose estimation. Although the controller uses hysteresis and distance thresholds to reduce chatter, residual bias or delay in the human model would translate directly into conservative or, if misaligned, optimistic clearance estimates. The capsule set is likewise an approximation; link radii and joint placements reflect the scene asset rather than precise anthropometrics.

6.5.3 Human variability

The human approach pattern in S2/S4 is a single sequence with one actor and one path of encroachment. It does not span different reach speeds, orientations, occlusions, bimanual gestures, or diverse body sizes. As a result, the reported dwell compliance and minimum-distance margins demonstrate that the gate logic works for the tested pattern, not that it is exhaustive over human behaviors. Broader coverage would require multiple trajectories, live streaming from a depth camera, and stress tests for discontinuities and occlusions.

6.5.4 Scene-specific tuning

Gains and thresholds, e.g., LSPB speed/acceleration limits, damping levels, posture weights, STOP r_{stop} and RELEASE r_{rel} radii, and the null-space smoothing factor were selected for the present scene (table height, tool posture, approach direction). Different fixtures, payloads, or tasks may change manipulability, available clearance, or visual occlusion, and hence call for different values. While the architecture is modular, its performance envelope is tied to these settings.

6.5.5 Unmodeled dynamics

The analysis assumes free-space motion with no external contacts beyond the virtual proximity field. Cable drag, joint backlash, gripper compliance, and flexible tools are not represented. In such conditions the measured equality residuals and

leakage bounds could degrade; feasibility might still hold (as in S5) but with tighter rate caps or stronger damping than used here.

6.5.6 Safety margins and conservative choices

The controller favors predictability and constraint satisfaction over raw throughput: STOP radii and dwell times are set to produce unambiguous freezes and smooth resumptions; strict null-space projection eliminates task contamination at the cost of reduced secondary authority near kinematic singularities. These choices are appropriate for shared workspaces but are not unique; a different risk budget could legitimately trade aggressiveness for throughput.

In summary, the evidence supports the intended behaviors of the proposed architecture under the tested conditions, but it should not be over-generalized. A complete validation would include hardware trials at the target control period, calibration-aware human tracking with latency accounting, multiple human approach patterns, and parameter sweeps under different tools and fixtures.

6.6 Reproducibility and data/code availability

All experiments in Section 6.1 are generated from archived scripts and logs. Each scenario bundle includes: configuration snapshot (with RNG seeds), run_uid, CSV logs (states, distances, modes), and figure/table exports. Reproduction requires (i) CoppeliaSim with the Panda scene used in this chapter, (ii) MATLAB for the controller scripts, and (iii) the skeleton data file for human pose where applicable. Runs are deterministic given the configuration and seed. The checklist below records, per scenario, the scene file, entry script, configuration and key parameters, human-data dependency, seed, the exact logs/CSV exports used to generate Chapter-6 figures and tables, and the repository commit/tag. Figures and tables can be regenerated by rerunning the listed script with the corresponding configuration.

Here we have table 6.17 of reproducibility checklist (per scenario)

Scenario	Scene file (.ttt)	Entry script	Config file / key params	Human data	Seed	Logs / CSV exports	Figures / Tables (IDs)	Repository reference
S1 — Base1 (vector attractive)	Base_1&2.ttt	S1_base_vector.m	config_S1.m (speed cap; posture weights)	None	42	S1_time_history.csv; S1_caps.csv	Figs 6.1–6.6; Table 6.3	Local archive (Chapter-6 bundle)

S2 — Master1 (vector + human)	Master_1 &2.ttt	S2_master_vector_human.m	config_S2.m (r_stop=0.25 m; r_rel=0.28 m; dwell=0.25 s)	skeleton_data3_icinco22.mat	42	S2_distances.csv; S2_state_timeline.csv	Figs 6.7–6.15; Table 6.5	Local archive (Chapter-6 bundle)
S3 — Base2 (LSPB)	Base_1&2.ttt	S3_base_LSPB.m	config_S3.m (LSPB v_max; a_max)	None	42	S3_speed_profile.csv; S3_kappa_sigma.csv	Figs 6.16–6.22; Table 6.7	Local archive (Chapter-6 bundle)
S4 — Master2 (LSPB + human)	Master_1 &2.ttt	S4_master_LSPB_human.m	config_S4.m (LSPB v_max; a_max; r_stop=0.25 m; r_rel=0.28 m; dwell=0.25 s)	skeleton_data3_icinco22.mat	42	S4_state_timeline.csv; S4_conditioning.csv	Figs 6.23–6.30; Tables 6.8–6.9	Local archive (Chapter-6 bundle)
S5 — Master3 (fixed TCP, null-space safety)	Master_3.ttt	S5_fixedTCP_nullspace.m	config_S5.m (RIF_STOP= 0.25 m; RIF_RELEASE=0.28 m; LEAK_THR =1e−6; dt=5 ms)	skeleton_data3_icinco22.mat	42	S5_TCP_lock.csv; S5_per_joint_motion.csv; S5_distance_feasibility.csv; S5_constraint_bindings.csv	Figs 6.31–6.33; Tables 6.10–6.13	Local archive (Chapter-6 bundle)

Table 6.17 Reproducibility checklist for Chapter 6. Scene files in some cases differ with respect to each scenario; CSVs are exported by each script's post-simulation step in outputs.

Chapter 7

Discussion in the Context of the Literature

This chapter situates the Chapter-6 evidence—obtained with a unified LSPB–DLS–SVD controller under explicit SSM supervision—within the HRC literature, focusing on what most directly governs safe, legible bench-top collaboration: null-space containment, SSM dwell semantics, and controller-rate proximity signals:

- How safety actions are confined to the Jacobian null space so that tool-level objectives are preserved (null-space compliance/containment);
- How explicit speed-and-separation monitoring (SSM) with hysteresis and dwell governs approach, pause/stop, and release;
- How perception outputs are reduced to controller-rate proximity signals (skeleton-to-capsule distances and link proxies) that are fast, smooth, and sensor-agnostic.

The discussion is anchored in the staged scenarios of Chapter 6. Two scenarios provide baselines (vector-attractive tracking without a person; LSPB tracking with bounded acceleration/jerk), two probe SSM behavior under person proximity (pause/resume repeatability; threshold dwell), and one fixes the TCP while using only redundancy to reshape posture and enlarge clearance. These configurations were executed in a synchronized MATLAB–CoppeliaSim loop so that reference sampling, Jacobian evaluation, projection, and integration share a common clock and are logged with control-tick timestamps. We interpret each observed behavior relative to literature that (i) injects avoidance through additive partial Jacobians with null-space projection on UR-class arms, (ii) implements collaborative-cell SSM with multi-camera tracking at ~30 Hz, and (iii) artificial potential field (APF)-style path-shaping for predictable side-choice. We emphasize that additive Jacobian terms without strict projection can contaminate the task—precisely what our leak guard prevents.

Three guiding questions structure the chapter. First, to what extent do the results show strict containment of safety-motivated reconfiguration—i.e., avoidance in the

null space without measurable leakage into the task channel—and how does this compare to additive partial-Jacobian schemes reported for UR-family robots? This matches the containment targets advocated by recent null-space compliance papers and contrasts with additive partial-Jacobian blending, which risks task leakage if projection is not enforced [5, 6, 7]. Second, are pause/stop/release transitions reproducible and legible (no chattering, consistent dwell), as required by SSM practice in collaborative assembly cells with continuous human access? (see the metric dictionary and scenario logs for dwell counters, STOP/RELEASE timestamps, and restart smoothness). Third, does reducing perception to lightweight geometric surrogates (skeleton-derived capsules; link proxies) achieve the intended controller-rate stability without sacrificing responsiveness relative to multi-view point-cloud fusion pipelines? Capsules deliver closed-form distances at control-rate, whereas composite SDFs trade higher fidelity for cycle-time budget; both are consistent with recent fast-collision-checking result [9, 11]. We answer each using your measured indicators—TCP drift and orientation lock near targets, minimum-distance timelines, state-transition logs, singular values/condition numbers, and saturation flags—which were selected to expose both numerical health and human-facing legibility.

A final thread concerns predictability of the robot's path around people. APF-based methods often trade analytical elegance for unpredictable detours near obstacles; the "local attractor" refinement bends trajectories to enforce a priori side-choice without introducing local minima. Although the present controller is not APF-driven, the same user-facing property is achieved procedurally—via bounded-jerk LSPB commands, explicit SSM thresholds with dwell, and null-space posture shaping—so the tool motion remains legible while clearance grows through redundancy. The fixed-TCP scenario (S5) is the critical stress test: the tool pose remains effectively invariant while joints reconfigure to increase separation, demonstrating kinematic "invisibility" of safety actions at the TCP and thereby satisfying the strictest interpretation of task preservation.

7.1 Null-space compliance, containment, and tracking integrity

In Scenario S5 (fixed-TCP), the commanded tool pose is held constant while redundancy alone reshapes posture in response to proximity. The measured TCP

translation remains sub-millimetric with zero orientation drift; equality residuals at the TCP are at numerical precision and no constraint class binds. These data indicate strict containment of safety actions within the null space and a stable SVD-regularized DLS inversion under supervision.

This behavior operationalizes the separation advocated in task-priority control: primary tool objectives are preserved while secondary behaviors (posture shaping, joint-limit avoidance, and avoidance biases) are confined to the Jacobian's null space. UR-series exemplars compute repulsive operational-space velocities at link points, map them through partial Jacobians, and project the summed avoidance term with $(I - J^{+}J)$ so the primary task continues when feasible—your stack generalizes the concept with SVD-regularized DLS and unified time-law generation.

In the broader literature on null-space compliance variation, safety is sometimes traded against tracking by altering compliance in redundant directions; your fixed-TCP results demonstrate the opposite extreme—zero-leakage at the TCP—consistent with a design that prioritizes supervisor-level modulation (pause/hold) over blending safety fields into the task channel.

7.2 Explicit governors, SSM, and dwell semantics

supervisor implements a clear approach–caution–pause–stop–release ladder with hysteresis and dwell. Chapter 6 logs show single, crisp STOP/RELEASE sequences, no chattering at thresholds, and reproducible resumption from consistent states. This mirrors collaborative-cell implementations where multi-view human tracking (~30 Hz) supports responsive collaboration while the robot yields predictably under SSM.

Design-wise, collision avoidance is governed rather than free-running: the controller preserves tool intent via null-space shaping; the supervisor arbitrates progression and holds; perception provides only the proximity signals needed to keep transitions auditable. This separation of concerns is consistent with industrial collaborative layouts and contributes to the legibility observed in your pause/resume scenarios.

7.3 Capsule and distance pipelines versus point-cloud fusion

Your perception path converts skeleton keypoints into limb-aligned capsules; robot links are paired with simple proxies; minimum distances between selected limb–link pairs are evaluated each control tick and rate-limited before feeding the supervisor and null-space shaper. The choice privileges timing and smoothness over raw fidelity and is consistent with HRC reports that either (a) fuse multiple depth views into a higher-fidelity point cloud and then compute distances, or (b) operate directly on lightweight geometric abstractions for controller-rate stability. In both cases, the acquisition/processing loop commonly runs at the camera update rate (~30 Hz) while the controller runs faster.

Point-cloud fusion pipelines with two Kinect v2 devices and a dual-PC architecture demonstrate practical latency management and real-time distance computation under repulsive control; your MATLAB–CoppeliaSim synchronization and logging regime adopt the same ethos—favoring determinism and observability—while keeping the control loop agnostic to the particular sensor brand or SDK.

UR-family experiments further validate distance-driven repulsion mapped through partial Jacobians and projected to null space; your results extend this doctrine to a 7-DoF Panda, coupled with explicit SSM gating and a unified LSPB time law for legible tracking.

7.4 APF with local attractors, predictability, and strict containment

Classical APF methods can yield path unpredictability near obstacles. The “local attractor” formulation bends the field so that the robot passes on a chosen side, while avoiding additional local minima—studied in theory and validated experimentally for mobile robots by tuning intensity and decay to balance predictability with curvature.

Although the present system is not APF-driven, it achieves equivalent user-facing predictability procedurally: bounded-jerk LSPB references, null-space posture shaping, and SSM dwell encode where and how motion proceeds or yields. In fixed-TCP runs, safety actions become kinematically invisible at the tool, achieving predictability without field-induced task leakage.

7.5 Comparative positioning

Table 7.1 demonstrates how the proposed LSPB–DLS–SVD framework surpasses representative approaches by enforcing strict null-space containment under SSM, preserving task guarantees, and delivering legible, predictable motion with a streamlined perception architecture.

Axis	APF + Repulsion (UR3)	APF w/ Local Attractors	Collaborative Cell (2×Kinect)	This work (LSPB–DLS–SVD + SSM)
Safety mechanism	Repulsive velocities at link points → partial JGi → $(I-J^{+}J)$ projection	Side-choice via shaped field without new minima	SSM; responsive collaboration with continuous access	SSM gating + strict null-space containment
Control layer	Additive partial-Jacobian repulsion; null-space projection	Field shaping (local planner)	Task logic + online avoidance; ~30 Hz tracking	Task-priority IK (SVD-DLS) + LSPB; safety in null space
Redundancy use	Exploited for link avoidance	Not explicit	Implicit, system-dependent	Primary: absorb avoidance; fixed-TCP option
Perception	MoCap/depth → distances at control rate	Not specific (mobile demo)	Two Kinect v2; multi-PC; TCP/IP	Skeleton→capsules; link proxies; synchronized logs
Task guarantees	May degrade if projection conflicts	Goal reaching may oscillate near obstacles	Throughput with continuous human access	Measured zero leakage at TCP; clean STOP/RELEASE
Legibility	Emergent, geometry-dependent	A priori side-choice via attractors	Predictable pauses/resumes	Predictable via SSM dwell + LSPB

Table 7.1 Comparison of representative HRC motion/safety strategies vs. this work. Columns summarize (i) safety injection locus, (ii) task-leakage risk, (iii) pause/stop/resume semantics, and (iv) legibility/reproducibility under SSM dwell.

7.6 Contribution summary

Relative to the above, this work contributes a unified, experimentally validated stack that:

• Demonstrates strict null-space containment under explicit SSM supervision, including a fixed-TCP mode where safety reconfiguration is kinematically invisible at the tool (cf. S5 TCP-lock logs and leakage residuals).

• Couples bounded-jerk LSPB references with SVD-regularized DLS to preserve legibility and numerical robustness while logging conditioning, residuals, and constraint activation for auditability.

• Integrates a skeleton-to-capsule distance pipeline and synchronized MATLAB–CoppeliaSim logging to support reproducible SSM dwell behavior and side-choice predictability without resorting to field shaping.

7.7 Limitations and scope

Local-planner studies note that strong shaping or high curvature near obstacles can saturate actuators and degrade tracking. “Local-attractor” methods mitigate this by bending trajectories without creating new minima. Our architecture avoids this fragility by offloading predictability to supervisory logic and keeping control declarative (task vs. null space). Nonetheless, results are simulation-centric: transferring STOP/RELEASE equality at the TCP and distance-rate conditioning to hardware will require tighter sensing latencies and middleware with deterministic timing; multi-PC Kinect layouts and UR-class external control demonstrate feasibility.

7.8 Concluding synthesis and lead-in to Chapter 8

Chapter 6 shows that a supervisor-first design can deliver predictability and strict task preservation via null-space shaping—outcomes that APF variants achieve through field design, here realized architecturally. The literature supports each pillar independently (null-space projection for redundancy resolution; SSM with dwell for reproducible behavior; multi-view or capsule-based distance for robust, low-latency inputs). The principal contribution is to demonstrate that a unified LSPB–DLS–SVD controller with explicit SSM and a lightweight capsule pipeline can jointly deliver zero-leakage tracking and predictable yielding/resumption. Chapter 8 now formalizes these contributions (8.1) and lays out the hardware-credible roadmap (8.2).

Chapter 8

Contributions & Future Work

This chapter distills what the thesis has achieved and outlines a concrete path forward. The central result is a system-level architecture that preserves tool-level intent while managing human–robot clearance through redundancy, under explicit speed-and-separation supervision. We first summarize these contributions in a compact form (Section 8.1), then identify the most impactful extensions toward hardware deployment and increased formal safety guarantees (Section 8.2).

8.1 Contributions

This thesis delivers an implementation-level control and supervision stack for collaborative manipulation that preserves tool-level intent while managing human–robot clearance through redundancy, with behaviors that are legible, repeatable, and auditable. The main contributions are:

- **Unified LSPB–DLS–SVD controller**

  A single task-priority layer executes either bounded-jerk LSPB references or vector-attractive commands through an SVD-regularized damped least-squares IK. This keeps responses well-conditioned near singularities and joint limits while maintaining consistent transient behavior across scenarios (benchmarked in S1–S4 conditioning and restart smoothness).

- **Strict null-space containment of safety actions**

  Safety-motivated posture regulation (repulsion, joint-limit avoidance, posture shaping) is confined to the Jacobian null space so the commanded tool motion remains intact whenever redundancy allows. In the fixed-TCP configuration, the TCP pose remains effectively invariant while joints reconfigure to enlarge clearance (zero leakage at the task). We enforce $\|J\,N(q)\,\dot{q}rep\| \le \varepsilon\_leak$ ($\varepsilon\_leak = 10^{-6}$) in logs.

- **Explicit SSM supervision with calibrated dwell**

  Approach–caution–pause–stop–release are governed by thresholds, hysteresis, and dwell times chosen for non-chattering behavior at boundaries. STOP/RELEASE is reproducible and returns the controller to a consistent state, supporting auditability and operator trust; dwell counters and STOP/RELEASE timestamps are stored per tick.

- **Geometry-first perception to controller-rate distances**

  Human pose streams are converted into limb-aligned capsules; robot links are paired with lightweight proxies. Minimum distances on selected limb–link pairs are debounced and rate-limited each control tick, yielding smooth, low-latency proximity cues to both the supervisor and the null-space shaper while remaining sensor-agnostic. Capsules preserve cycle budget; composite SDFs remain an interchangeable higher-fidelity option.

- **Discrete-time correctness and logging alignment**

  Reference sampling is tied to the physics step; projection and smoothing precede integration. All quantities (poses, distances, Jacobians, singular values, manipulability, saturation flags, supervisor modes) are time-stamped at control-tick granularity, enabling replayable experiments and clear failure surfaces.

- **Fixed-TCP reconfiguration as a safety primitive**

  When task progression is not permitted, the arm increases separation purely through redundancy while holding the tool pose. This isolates safety behavior from task execution, clarifies operator expectations, and provides a conservative fallback without sacrificing legibility (S5: TCP-lock traces and per-joint motion logs).

- **Staged evaluation suite for HRC behaviors**

  A five-scenario progression probes baseline tracking, supervised pause/resume under proximity, time-parameterized motion with deterministic restart, and fixed-TCP posture reshaping. Common metrics—

TCP error, minimum-distance timelines, mode transition histories, conditioning indicators—enable like-for-like comparisons and ablations.

- **Transferable, simulator-synchronized testbed**

  A MATLAB–CoppeliaSim workflow ensures consistent geometry, frames, units, and timing across runs. The artifacts (code, logs, plots) form a reusable template for extending the approach to other redundant manipulators and sensing stacks.

8.2 Future Work

The following extensions prioritize hardware credibility, formal safety envelopes, and richer proximity modeling, while preserving the architecture's clarity and legibility.

- **Null-space compliance variation (safety–tracking trade-offs)**

  Introduce programmable compliance in redundant directions to adapt conservativeness online (e.g., higher stiffness for tracking when far from people; lower stiffness near people). Retain strict projector use and enforce an online residual cap ($\|r\| \leq \varepsilon\_leak$) to guarantee no task leakage while modulating compliance.

- **Certified Reference Governors (explicit envelopes)**

  Layer an explicit reference governor (ERG) above the supervisor to certify that commanded references remain inside provable distance/velocity bounds before execution. Use measured dwell/threshold behavior to calibrate ERG margins, and log governor interventions as auditable events; record governor activations with pre/post reference and active constraints.

- **Composite signed-distance fields (SDFs) for articulated robots**

  Replace capsule-only distances with composite SDFs that maintain controller-friendly gradients and handle complex shapes. Start with an offline SDF bake of robot links and "thickened" human limb models; deploy runtime queries that remain within current cycle budgets , targeting ≤ one control-tick per query.

- **Hardware-in-the-loop and ROS 2 migration**

  Port the synchronized loop to ROS 2 with deterministic executors (real-time rclcpp).

  Validate on an actual Panda/FR3:

  – replicate S1–S5;

  – verify STOP/RELEASE equality at TCP on hardware;

  – profile latencies (sensor → supervisor → joint command);

  – exercise loss/recovery of pose streams (dropouts, mis-detections).

- **Automatic dwell and threshold tuning**

  Close the loop on SSM parameters by optimizing dwell/thresholds against measured chattering rate, false stops, and resume lag. Use replayed human traces and multi-objective search (minimize stop count, maximize minimum distance, cap cycle-time overhead).

- **Learned postural priors with safety filters**

  Train light postural priors (e.g., manipulability-aware or ergonomics-aware secondary objectives) and filter them through the null-space projector with barrier terms for joint limits and clearance. Keep learning out of the task channel; log all activations.

- **Multi-person and tool/workpiece modeling**

  Extend the capsule set to multiple people and include tool/workpiece proxies. Prioritize limb–link pairs by risk and visibility; keep the controller load constant by capping active pairs per tick.

- **Formal verification and runtime monitors**

  Specify supervisor and projector properties in temporal logic (e.g., “no TCP displacement above ε during STOP”). Build runtime monitors that flag violations, snapshot the state, and support post-mortem analysis.

References

[1] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: Modelling, Planning and Control*. Springer, London, 2009. ISBN 978-1-84628-641-4.

[2] Kevin M. Lynch and Frank C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017. ISBN 978-1107156302.

[3] Alexander Reiter. *Optimal Path and Trajectory Planning for Serial Robots*. Springer, 2022. ISBN 978-3-658-28593-7.

[4] Michael A. Goodrich and Alan C. Schultz. "Human–Robot Interaction: A Survey." *Foundations and Trends in Human–Computer Interaction* 1(3): 203–275, 2007/2008. doi:10.1561/1100000005.

[5] Julian M. S. Ducaju, Björn Olofsson, Anders Robertsson, and Rolf Johansson. "Null-Space Compliance Variation for Safe Human–Robot Collaboration in Redundant Manipulators Using Safety Control Barrier Functions." In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 1–8. (Lund University portal preprint.)

[6] Fan Yang, Jiaqi Wang, and K. K. Kermani. "A Null Space Compliance Approach for Maintaining Safety and Tracking Performance in Human–Robot Interactions." arXiv:2502.02443, 2025.

[7] Kelly Merckaert, Bryan Convens, Chi-Ju Wu, Alessandro Roncone, Marco M. Nicotra, and Bram Vanderborght. "Real-Time Motion Control of Robotic Manipulators for Safe Human–Robot Coexistence." *Robotics and Computer-Integrated Manufacturing* 73 (2022): 102223. doi:10.1016/j.rcim.2021.102223. (Author PDFs: VUB/HIRO group.)

[8] Kelly Merckaert. *Certified Safe, Fast, and Real-Time Robot Control in Workspaces Shared by Humans and Robots*. Ph.D. Thesis, Vrije Universiteit Brussel, 2023.

[9] Baolin Liu, Gedong Jiang, Fei Zhao, and Xuesong Mei. "Collision-Free Motion Generation Based on Stochastic Optimization and Composite Signed Distance Field Networks of Articulated Robot." arXiv:2306.04130, 2023.

[10] Juan Calzada-García, José G. Victores, Francisco J. Naranjo-Campos, and Carlos Balaguer. “A Review on Inverse Kinematics, Control and Planning for Robotic Manipulators With and Without Obstacles via Deep Neural Networks.” *Algorithms* 18(1): 23, 2025. doi:10.3390/a18010023.

[11] Ziyang Xie, Lu Lu, Hanwen Wang, Li Li, and Xu Xu. “An Image-Based Human–Robot Collision Avoidance Scheme: A Proof of Concept.” *IISE Transactions on Occupational Ergonomics and Human Factors* 12(1–2): 112–122, 2024. doi:10.1080/24725838.2023.2222651.

[12] Alexander J. Elias and John T. Wen. “Redundancy Parameterization and Inverse Kinematics of 7-DOF Revolute Manipulators.” *Mechanism and Machine Theory* 204 (2024): 105613. (See also arXiv:2307.13122.)

[13] *Integrated Optimisation of Human–Robot Collaborative Disassembly*, Taylor & Francis, 2024.

[14] S. Mielke et al., “Human–Robot Planar Co-manipulation,” *Frontiers in Robotics and AI*, 2024.

[15] W. Xia et al., “Towards Human Modeling for HRC and Digital Twin,” 2025.

[16] L. S. Scimmi, M. Melchiorre, M. Troise, S. Mauro, and S. Pastorelli, “A Practical and Effective Layout for a Safe Human-Robot Collaborative Assembly Task,” *Applied Sciences*, vol. 11, no. 4, p. 1763, Feb. 2021. doi:10.3390/app11041763.

[17] M. Melchiorre, L. S. Scimmi, L. Salamina, S. Mauro, and S. Pastorelli, “Experiments on the Artificial Potential Field with Local Attractors for Mobile Robot Navigation,” *Robotics*, vol. 12, no. 3, p. 81, Jun. 2023. doi:10.3390/robotics12030081.

[18] M. Melchiorre, L. S. Scimmi, L. Salamina, S. Mauro, and S. Pastorelli, “Robot Collision Avoidance based on Artificial Potential Field with Local Attractors,” in *Proc. 19th Int. Conf. on Informatics in Control, Automation and Robotics (ICINCO)*, 2022, pp. 340–350. doi:10.5220/0011353200003271.

[19] M. Melchiorre, L. S. Scimmi, S. P. Pastorelli, and S. Mauro, “Collision Avoidance using Point Cloud Data Fusion from Multiple Depth Sensors: A Practical Approach,” *IEEE conference paper*, 2019.

[20] L. S. Scimmi, M. Melchiorre, S. Mauro, and S. P. Pastorelli, “Implementing a Vision-Based Collision Avoidance Algorithm on a UR3 Robot,” *IEEE conference paper*, 2019.

[21] L. S. Scimmi, M. Melchiorre, S. Mauro, and S. Pastorelli, “Multiple Collision Avoidance between Human Limbs and Robot Links Algorithm in Collaborative Tasks,” in *Proc. 15th Int. Conf. on Informatics in Control, Automation and Robotics (ICINCO)*, 2018, pp. 291–298.

[22] O. Khatib, “Real-Time Obstacle Avoidance for Manipulators and Mobile Robots,” *Int. J. Robot. Res.*, vol. 5, no. 1, pp. 90–98, 1986.

[23] C. I. Connolly and R. A. Grupen, “On the Applications of Harmonic Functions to Robotics,” 1993.

[24] D. Fox, W. Burgard, and S. Thrun, “The Dynamic Window Approach to Collision Avoidance,” 1997.

[25] A. Chakravarthy and D. Ghose, “Obstacle Avoidance in a Dynamic Environment: A Collision Cone Approach,” *IEEE Trans. Syst., Man, Cybern.*, 1998.

[26] E. Rimon and D. E. Koditschek, “Exact Robot Navigation using Artificial Potential Functions,” *IEEE Trans. Robot. Autom.*, 1992.

[27] R. Volpe and P. Khosla, “Manipulator Control with Superquadratic Artificial Potential Functions,” 1990.

[28] L. De Medio and G. Oriolo, “Tangential Fields for Path Shaping,” 1991.

[29] R. Murphy, “Selective Attraction for Predictable Paths,” 2000.

[30] A. Arslan and E. R. Koditschek, “Sensor-Based Reactive Navigation in Unknown Convex Sphere Worlds,” 2019.

[31] M. Castelnovi, A. Sgorbissa, and R. Zaccaria, “Projecting Goals to Overcome Local Minima,” 2006.

[32] I. Paromtchik and R. Nassal, “Temporary Goal Projection in APF,” 1995.

[33] Cosmin Pozna, Thomas Troester, Radu-Emil Precup, László Tar, and Stefan Preitl. “On the Design of an Obstacle Avoiding Trajectory: Method and Simulation.” *Mathematics and Computers in Simulation* 79(7): 2211–2226, 2009. doi:10.1016/j.matcom.2008.12.015.

[34] Ioannis Filippidis and Kostas J. Kyriakopoulos. “Adjustable Navigation Functions for Unknown Sphere Worlds.” In: *Proc. 50th IEEE Conference on Decision and Control (CDC)*, 2011, pp. 4276–4281.

[35] Jörg Güldner and Vadim I. Utkin. “Tracking the Gradient of Artificial Potential Fields: Sliding Mode Control for Mobile Robots.” *International Journal of Control* 63(3): 417–432, 1996. doi:10.1080/00207179608921850.

[36] S. Mauro et al., “Distance-to-Velocity Mappings for Repulsion,” 2017.

[37] Martina Melchiorre, Lorenzo S. Scimmi, Simone Mauro, and Stefano Pastorelli. “A Novel Constrained Trajectory Planner for Safe Human-Robot Collaboration.” In: *Proc. 19th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2022, pp. 287–294.

[38] Lorenzo S. Scimmi, Martina Melchiorre, Simone Mauro, and Stefano Pastorelli. “Multiple Collision Avoidance between Human Limbs and Robot Links: Algorithm in Collaborative Tasks.” In: *Proc. 15th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2018, Vol. 2, pp. 291–298.

[39] Lorenzo S. Scimmi, Martina Melchiorre, Simone Mauro, and Stefano Pastorelli. “Implementing a Vision-Based Collision Avoidance Algorithm on a UR3 Robot.” In: *Proc. 23rd International Conference on Mechatronics Technology (ICMT)*, Salerno, Italy, Oct 23–26, 2019, pp. 1–6.

[40] Lorenzo S. Scimmi, Martina Melchiorre, Massimiliano Troise, Simone Mauro, and Stefano Pastorelli. “A Practical and Effective Layout for a Safe Human-Robot Collaborative Assembly Task.” *Applied Sciences* 11(4): 1763, 2021. doi:10.3390/app11041763.

[41] ISO/TS 15066:2016. *Robots and Robotic Devices — Collaborative Robots*. International Organization for Standardization (ISO), Geneva, 2016.

[42] Peter I. Corke. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. 2nd ed., Springer, 2017. (Robotics Toolbox reference.)

[43] ISO 10218-1:2011. *Robots and Robotic Devices — Safety Requirements for Industrial Robots — Part 1: Robots*. ISO, Geneva, 2011.

[44] ISO 10218-2:2011. *Robots and Robotic Devices — Safety Requirements for Industrial Robots — Part 2: Robot Systems and Integration*. ISO, Geneva, 2011.

[45] Kevin M. Lynch and Frank C. Park. "Modern Robotics — Resource/Preprint Site." Northwestern University.