CLOSED-FORM ACTION PROJECTION FOR CONSTRAINED WORLD MODEL PLANNING IN ROBOTICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Physical control systems face hard constraints at every manipulation step, such as strict bounds on contact force and velocity. However, existing reward-driven reinforcement learning struggles to enforce these limits. This challenge is compounded in practice because constraint satisfaction depends not only on the high-level predicted action, but also on the low-level controller's live state during execution. To bridge this gap, we introduce **CAP**$^{\textbf{O}}$ (Closed-form Action Projection), a modular layer that imposes hard, state-dependent action constraints on learned policies and world-model planners. At each control step, **CAP**$^{\textbf{O}}$ derives an admissible action set in closed form from the live controller state and projects the proposed action onto it without any online optimization or a learned constraint model, provably guaranteeing constraint satisfaction. To resolve the geometric ambiguity inherent in pure proprioception, we integrate a world model that processes environmental observations, such as wrist depth, during impedance control. Evaluations on three representative contact-rich manufacturing tasks, peg insertion, gear meshing and nut threading, demonstrate that **CAP**$^{\textbf{O}}$ strictly prevents force limit violations at every step while maintaining high task success rates, achieving significant improvement against the state-of-the-art neural baselines.

Videos, models, and code: cap-wm.github.io

1 INTRODUCTION

Safety is a central requirement for robots that interact with people or manipulate delicate objects (Svarny et al., 2019; Matschek et al., 2023). In assembly, contact is both necessary for completing the task and a potential source of damage: inserting a connector, meshing gears or threading a nut requires controlled interaction under uncertainty about part pose and geometry (Noseworthy et al., 2025). The challenge is to learn effective contact behavior while respecting a prescribed force limit throughout execution.

Reinforcement learning has enabled assembly policies that transfer from simulation to physical robots (Tang et al., 2023; Noseworthy et al., 2025). World models support planning through learned predictions (Wu et al., 2022; Hansen et al., 2024), and wrist-camera depth extends their use to unfamiliar insertion geometries (Hansen et al., 2026). However, task success does not establish force compliance: a reward penalty allows excessive contact to be traded against task progress. Constrained reinforcement learning addresses this trade-off by bounding expected cumulative cost (Achiam et al., 2017; Stooke et al., 2020). A positive cost budget can still admit brief force violations, whereas force-limited manipulation requires a cap at each control step.

Enforcing this cap requires accounting for the controller that executes the action. A learned planner commands a pose target, which an impedance controller converts to a wrench using its gains, robot motion, action-filter memory, and actuation dead zone. The same action can therefore produce different forces across controller states, so a fixed action clamp does not express a fixed force limit. We ask: *How do we bound the force a controller applies, rather than the action a policy commands, at every step of a learned planner?*

We introduce **Closed-form Action Projection (CAP**$^{\textbf{O}}$**)**, which derives admissible action intervals from live controller state and projects the proposed action by coordinate-wise clipping. The intervals combine force and speed limits with target clipping, accounting for smoothing and dead zones

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

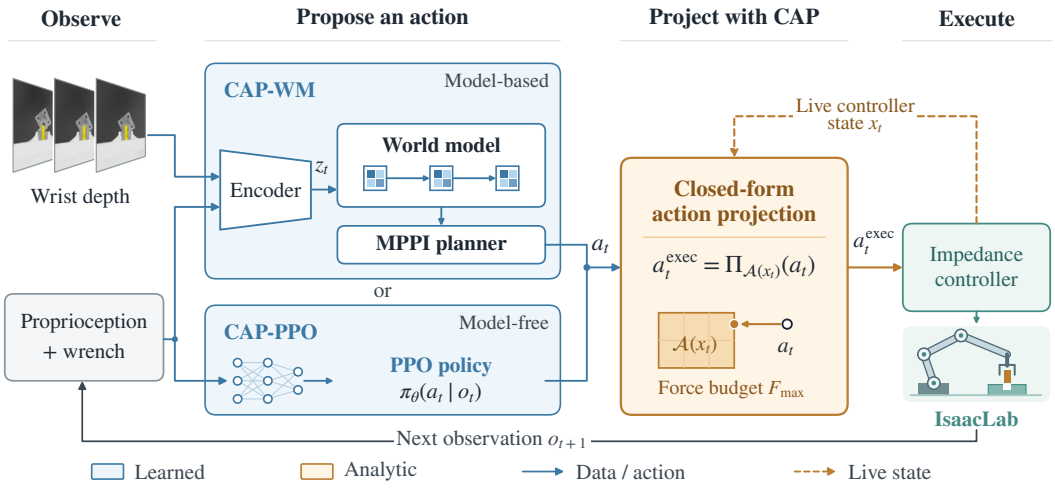

Figure 1: **Architecture of the two CAP agents.** CAP-WM plans from wrist depth, proprioception, and wrench observations; CAP-PPO proposes actions from proprioception and wrench observations. Both feed the same projection, which uses the live controller state x_t to map the proposal a_t into the admissible set $\mathcal{A}(x_t)$ before impedance control. Solid arrows carry observations or actions; dashed arrows carry controller state. The camera thumbnails illustrate the scene; only CAP-WM receives depth.

without an online optimization solver or learned constraint model. CAP-WM couples this layer with a latent world model over wrist depth, proprioception, and force; CAP-PPO uses the same layer with a PPO policy (Figure 1). Depth supplies information about contact geometry, while projection remains active during data collection so the learner adapts to the constrained system. The applied-wrench guarantee holds when the interval intersection is nonempty; measured contact-force bounds additionally require the calibration conditions in Section 4. To the best of our knowledge, CAP-WM is the first framework to combine world-model planning with closed-form, controller-aware action projection for contact-rich manipulation.

Our contributions are threefold. First, we introduce CAP-WM, which combines world-model planning with closed-form, controller-aware action projection for contact-rich manipulation; the same projection also supports model-free learning through CAP-PPO. Second, we derive admissible action sets from the live controller state and prove per-step bounds on the applied wrench whenever these sets are nonempty, without online optimization or a learned constraint model. Third, we demonstrate on three simulated assembly tasks, peg insertion, gear meshing, and nut threading, that CAP-WM maintains high task success with no force-limit violations in the reported evaluations, substantially improving safe success over constrained RL and safety-layer baselines.

2 Related Work

Learning contact-rich manipulation. Learning-based assembly must accommodate geometric uncertainty while controlling physical interaction. IndustReal and AutoMate demonstrate sim-to-real transfer of precise assembly skills (Tang et al., 2023; 2024), and FORGE incorporates force feedback, threshold conditioning, and dynamics randomization to adapt contact behavior (Noseworthy et al., 2025). World models offer a predictive representation for learning such behaviors: DreamerV3 and DayDreamer learn through latent imagination (Hafner et al., 2025; Wu et al., 2022), while TD-MPC2 uses latent dynamics for online planning (Hansen et al., 2024). Combining these representations with wrist-depth observations supports insertion across unfamiliar geometries (Hansen et al., 2026). This line of work motivates the visual planning component of CAP-WM, alongside the force constraint imposed at execution.

Safe reinforcement learning and planning. Constrained RL formulates safety through additional costs, with CPO, PID-Lagrangian methods, and FOCOPS optimizing reward under expected cu-

Table 1: **Capability matrix.** Guarantee as each paper claims it: **Hard** = every step, by construction, relative to the paper's own model; Soft = reward penalty; Budget = expected episode cost; Chance = with a stated probability. Applied: the bound is on the wrench the controller applies. Closed form: no online solve. Learned model: plans or predicts with a learned model.

Method	Bounded quantity	Guarantee	Applied	Closed form	Learned model
FORGE (Noseworthy et al., 2025)	contact force (N)	Soft	×	×	×
CPO (Achiam et al., 2017)	dimensionless cost	Budget	×	×	×
SafeDreamer (Huang et al., 2024)	dimensionless cost	Budget	×	×	✓
SHaRe-RL (Stranghöner et al., 2025)	commanded force (N)	**Hard**	×	✓	×
Matschek et al. (2023)	contact force (N)	Chance	✓	×	✓
CAP🛡 (ours)	applied wrench (N)	**Hard**	✓	✓	✓

mulative cost budgets (Achiam et al., 2017; Stooke et al., 2020; Zhang et al., 2020). Model-based approaches use predicted outcomes to inform constrained behavior: SafeDreamer integrates safety-reward planning with Lagrangian objectives (Huang et al., 2024), while CASE combines constrained model-based learning with a planning-based backup filter (Agha et al., 2024), AMBS shields actions with world-model rollouts (Goodall & Belardinelli, 2024), and SafeCARL projects MPPI actions onto a conformal barrier constraint on learned control-affine dynamics (Zhou et al., 2026). Per-step constraints are distinguished from cumulative budgets in state-wise safe RL and robot learning (Zhao et al., 2024; Liu et al., 2021). CAP🛡 adopts the per-step requirement for force-limited manipulation and computes its execution constraint independently of learned cost predictions.

Constraint enforcement in robot control. Shielding and safety layers enforce constraints by restricting proposed actions before execution (Alshiekh et al., 2018; Krasowski et al., 2023; Markgraf et al., 2025). Constructions include analytic correction from learned local constraints (Dalal et al., 2018), tangent-space actions in ATACOM (Liu et al., 2021), and hard-constrained network outputs in HardNet (Min & Azizan, 2024). For physical interaction, the constrained quantity must be related to contact force: Matschek et al. (2023) use Gaussian-process force models within chance-constrained MPC, with recursive-feasibility analysis under their assumptions, while Guarding Force uses a force-constrained barrier function and quadratic programming (Wang et al., 2026). TACTIC scores contact forces in newtons from a hybrid learned-analytic model inside sampling-based MPC (Madan et al., 2026). Safe RL for contact-rich tasks pairs variable impedance with recovery policies or passivity layers (Zhang et al., 2024; 2026), and SHaRe-RL bounds forces through per-axis compliance (Stranghöner et al., 2025). Unlike these methods, CAP🛡 derives exact admissible intervals from the unmodified controller's action-to-wrench map, including smoothing and dead zones, and bounds the approach speed before contact, without an online solve or a learned force model (Table 1).

3 Preliminaries

3.1 Force-Limited Manipulation

We consider an episodic decision process in which, every Δt seconds, a learned agent observes o_t, applies an action $a_t \in \mathcal{A} = [-1, 1]^n$ to a low-level controller and receives a reward $R(s_t, a_t)$; the state $s_t \in \mathcal{S}$ evolves as $s_{t+1} \sim P(\cdot \mid s_t, a_t)$ and is seen only through o_t. A policy π maximizes the discounted return $J(\pi) = \mathbb{E}_\pi\big[\sum_t \gamma^t R(s_t, a_t)\big]$, $\gamma \in [0, 1)$. The quantity to be limited is the wrist force $F_t \in \mathbb{R}^3$, low-pass filtered by the environment and read at the decision instants, and the requirement is

$$\max_\pi \; J(\pi) \quad \text{subject to} \quad \|F_t\|_2 \le F_{\max} \;\; \text{at every step } t \text{ of every episode.} \tag{1}$$

The constraint in (1) is pointwise. It differs from the contract of constrained MDPs (Altman, 1999; Achiam et al., 2017), $\mathbb{E}_\pi[\sum_t \gamma^t c(s_t, a_t)] \le \bar{c}$, which bounds a discounted expectation and admits policies that exceed the limit on individual steps, and from the penalty on force above a threshold that the task reward of Noseworthy et al. (2025) contains. It also differs from a state constraint in where the constrained quantity comes from: F_t is produced by the controller from the action and

from its own state x_t (Section 3.2), so the limit constrains the action through the controller state, $a_t \in \mathcal{A}(x_t) \subseteq \mathcal{A}$, the input-dependent form of a hard constraint (Dalal et al., 2018; Min & Azizan, 2024). Enforcing it at every step enforces (1); Section 4 constructs $\mathcal{A}(x_t)$ in closed form.

3.2 Task-Space Impedance Control

The agent commands the Franka arm of FORGE (Noseworthy et al., 2025), built on Factory (Narang et al., 2022), through its task-space impedance controller, used unchanged. The action has seven components ($n = 7$): six pose-related commands and a success prediction. As in FORGE, however, control authority is restricted to translation and yaw, with roll and pitch held fixed. Each action is smoothed, $\tilde{a}_t = \lambda a_t + (1-\lambda)\,\tilde{a}_{t-1}$; the smoothed translational part sets a position target relative to the fixed part, the error to the target is clipped, a proportional-derivative law gives the task-space wrench, and a per-axis actuation dead zone d that emulates joint friction gives the applied wrench:

$$\begin{aligned} p_t^{\mathrm{d}} &= p^{\mathrm{ref}} + B\,\tilde{a}_{t,1:3}, \qquad W_t = K_p \operatorname{clip}\big(p_t^{\mathrm{d}} - p_t;\, \pm\delta\big) - K_d\, v_t, \\ W_t^{\mathrm{app}} &= D_d(W_t), \qquad D_d(w) = \operatorname{sign}(w)\, \max\big(0,\, |w| - d\big), \end{aligned} \tag{2}$$

with p_t, v_t the fingertip position and velocity and B, K_p, K_d, δ diagonal. The target is held for one control period, $\Delta t = 1/15\,\mathrm{s}$, while (2), its rotational counterpart and the conversion to joint torques run at $120\,\mathrm{Hz}$. Every episode redraws $\lambda \sim \mathcal{U}[0.025, 0.1]$, the gains K_p, the clip δ and the dead zone $d \sim \mathcal{U}[0, 5)\,\mathrm{N}$ per axis, the last again every $2\,\mathrm{s}$ (Appendix C.2); the agent observes none of them. The controller state $x_t = (K_p, K_d, \lambda, d, \delta, \tilde{a}_{t-1}, p_t, v_t, p^{\mathrm{ref}})$ thus determines, with the action, the wrench applied during the period.

3.3 Planning with a Latent World Model

CAP-WM🛡 builds on TD-MPC2 (Hansen et al., 2024), a decoder-free latent world model with parameters ϕ: an encoder $z_t = h_\phi(o_t)$, latent dynamics $z_{t+1} = f_\phi(z_t, a_t)$, a reward head $R_\phi(z, a)$, an ensemble of value heads $Q_\phi(z, a)$ and a policy prior $\pi_\phi(z)$, all multilayer perceptrons with SimNorm latents, trained on replayed sequences to predict the next latent, the reward and a TD value target. At every step TD-MPC2 optimizes an H-step action sequence to maximize the model-predicted return, with a terminal value standing in for the return beyond the horizon,

$$\max_{a_{t:t+H-1} \in \mathcal{A}^H} \sum_{k=0}^{H-1} \gamma^k R_\phi(z_{t+k}, a_{t+k}) + \gamma^H Q_\phi(z_{t+H}, a^\pi_{t+H}) \quad \text{s.t. } z_{t+k+1} = f_\phi(z_{t+k}, a_{t+k}), \tag{3}$$

where $a^\pi_{t+H} \sim \pi_\phi(\cdot \mid z_{t+H})$. The planner executes only the first action. (3) is solved by model predictive path integral control (MPPI; Williams et al., 2017), which refits a Gaussian over action sequences, seeded in part by the prior, to its highest-scoring samples for a fixed number of iterations. The planner optimizes over $\mathcal{A}$ and sees nothing of $\mathcal{A}(x_t)$; its output is the proposal that Section 4 projects.

4 Method

At every control step t a learned decision maker proposes an action $a_t \in \mathcal{A}$, and the wrench the arm applies depends on the pair (a_t, x_t) through (2). CAP🛡 derives from x_t, in closed form, an admissible set $\mathcal{A}(x_t) \subseteq \mathcal{A}$ and executes the projection $a_t^{\mathrm{exec}} = \Pi_{\mathcal{A}(x_t)}(a_t)$ of whatever is proposed (Figure 2). Section 4.1 constructs $\mathcal{A}(x_t)$ for force-limited manipulation; Sections 4.2 and 4.3 describe the world-model planner of CAP-WM🛡, our main instantiation, and the training of both it and the model-free CAP-PPO🛡 with the projection in the loop.

4.1 Closed-Form Action Projection

Affine wrench model. Where the clip in (2) is inactive, the smoothing and the proportional-derivative law compose to a wrench that is affine in the action on each translational axis i, and the applied wrench is its image under the dead zone:

$$W_{t,i}(a) = c_i(x_t) + m_i(x_t)\, a_i, \qquad W_{t,i}^{\mathrm{app}} = D_{d_i}\big(W_{t,i}(a)\big), \tag{4}$$

with $c_i(x_t) = K_{p,i}\big(p_i^{\mathrm{ref}} + B_i(1-\lambda)\,\tilde{a}_{t-1,i} - p_{t,i}\big) - K_{d,i}\,v_{t,i}$ and $m_i(x_t) = K_{p,i}B_i\lambda > 0$. All quantities in (4) are read from the controller at the decision instant, and none is learned.

Force interval. The bound is placed on the applied wrench: with a dead zone of several newtons, a cap on the commanded wrench would starve actuation. With a budget $\bar{F}_t \le F_{\max} - \varepsilon$ for the step and an allocation $\kappa \in \mathbb{R}^3_{>0}$, $\|\kappa\|_2 \le 1$ ($\kappa_i = 1/\sqrt{3}$ throughout),

$$\mathcal{I}_i^{\mathrm{force}}(x_t) = \Big\{a_i : \big|c_i + m_i a_i\big| \le d_i + \big(\kappa_i\,\bar{F}_t - K_{p,i}\,|v_{t,i}|\,\Delta t\big)_+\Big\}, \tag{5}$$

where Δt is the control period and $K_{p,i}|v_{t,i}|\Delta t$ absorbs the change of the position error while the arm moves during the period. By (4), every a_i in this interval gives $|W_{t,i}^{\mathrm{app}}| \le \kappa_i\bar{F}_t$ and hence $\|W_t^{\mathrm{app}}\|_2 \le \bar{F}_t$. The budget is tightened by the force measured at the previous step, $\bar{F}_t = \min\big(\bar{F},\,(\beta\,\bar{F} + (1-\beta)(F_{\max} - \|F_{t-1}\|))_+\big)$ with $\bar{F} = F_{\max} - \varepsilon$ and $\beta \in (0,1]$: the full budget is available while the previous reading stays within ε, and it decreases linearly as that reading grows. The margin ε covers the difference between the modelled applied wrench and the sensor reading and is calibrated per task from traces of separate calibration runs (Appendices B.2 and B.3).

Speed interval. A quasi-static bound cannot see impact: a rigid contact converts approach speed into a force spike within one control period, so the layer also bounds the speed at which the part may reach the mating surface. Over one period the arm responds to the applied wrench as a free body of task-space inertia Λ_i, so keeping its velocity $v_{t,i} + \Delta t\,W_{t,i}^{\mathrm{app}}/\Lambda_i$ after the step within a budget $\bar{v}_{t,i}$ confines the applied wrench to $[\ell_i, u_i] = \big[-\Lambda_i(\bar{v}_{t,i} + v_{t,i})/\Delta t,\ \Lambda_i(\bar{v}_{t,i} - v_{t,i})/\Delta t\big]$ and, through the dead zone and the affine map, the action to

$$\mathcal{I}_i^{\mathrm{vel}}(x_t) = \Big\{a_i : \ell_i \le D_{d_i}\big(c_i + m_i a_i\big) \le u_i\Big\}, \tag{6}$$

an interval because D_d is monotone (Lemma A.1). The budget $\bar{v}_t$ is a braking curve of the distance to the surface (Appendix B.3): it equals the touchdown speed $v_{\mathrm{touch}} = F_{\mathrm{imp}}/\sigma$ within one step of the surface, where σ is the calibrated force overshoot per unit approach speed and F_{imp} the share of the limit reserved for impact, and grows with the distance as the speed from which the controller can still brake to v_{touch}.

Projection. The clip in (2) makes (4) exact only where it is inactive; since $K_{p,i}\,(p_{t,i}^{\mathrm{d}} - p_{t,i}) = c_i + K_{d,i}\,v_{t,i} + m_i a_i$, a third interval, $\mathcal{I}_i^{\mathrm{clip}}(x_t) = \{a_i : |c_i + K_{d,i}\,v_{t,i} + m_i a_i| \le K_{p,i}\,\delta_i\}$, keeps it inactive by construction. Since $m_i > 0$, each of the three constraints is an interval in the single coordinate a_i, so the admissible set is a box and the Euclidean projection onto it is a componentwise clamp:

$$\mathcal{A}(x_t) = \prod_{i=1}^{3}\Big(\mathcal{I}_i^{\mathrm{clip}}(x_t) \cap \mathcal{I}_i^{\mathrm{vel}}(x_t) \cap \mathcal{I}_i^{\mathrm{force}}(x_t) \cap [-1,1]\Big) \times [-1,1]^{\,n-3}, \tag{7}$$

$$a_t^{\mathrm{exec}} = \Pi_{\mathcal{A}(x_t)}(a_t) = \mathrm{clip}\,\big(a_t;\,\mathcal{A}(x_t)\big). \tag{8}$$

The remaining coordinates are only clipped to the action range (on the gear task a kinematic keep-out interval on the yaw command, built the same way, keeps the fingers off the neighbouring gears; Appendix B.4). The projection costs a few arithmetic operations per axis and involves neither an online solve nor a learned constraint model. Input-dependent layers such as those of Dalal et al. (2018) and Min & Azizan (2024) also correct actions in closed form, for one linearized constraint or for affine constraints on the output; here the closed form follows from the controller itself, since each constraint involves a single action coordinate and the admissible set is therefore a box. The implementation clamps each coordinate through the three intervals and the action range in turn, which equals (8) whenever the box is nonempty (Lemma A.2). Because the force interval is applied last, the bound on the applied wrench also holds when the intervals of an axis do not intersect, on every step at which the target clip stays inactive and the final clip to the action range changes nothing (Corollary A.3).

Proposition 1 (Per-step force bound)**.** *Let $\mathcal{A}(x_t) \ne \emptyset$. For every proposed action $a_t \in \mathcal{A}$, the executed action $a_t^{\mathrm{exec}} = \Pi_{\mathcal{A}(x_t)}(a_t)$ leaves the target clip inactive, so that the commanded wrench is given by (4); the applied wrench satisfies $|W_{t,i}^{\mathrm{app}}| \le \kappa_i\,\bar{F}_t$ for $i = 1,2,3$ and hence $\|W_t^{\mathrm{app}}\|_2 \le \bar{F}_t \le F_{\max} - \varepsilon$; and the free-body velocity after the step satisfies $|v_{t,i} + \Delta t\,W_{t,i}^{\mathrm{app}}/\Lambda_i| \le \bar{v}_{t,i}$. The proof is in Appendix A.*

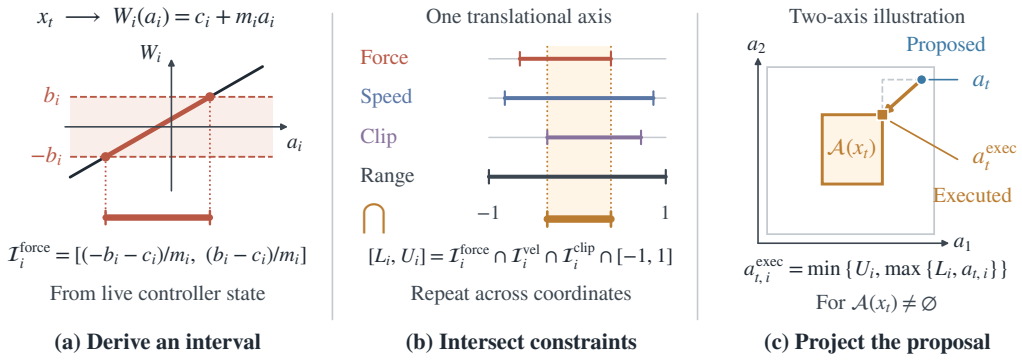

Figure 2: **Closed-form action projection on the translational axes.** (a) The affine wrench map converts an applied-force budget, adjusted for the dead zone, into an action interval on the unclipped controller branch. (b) Intersecting the force, speed, target-clip, and action-range intervals yields $[L_i, U_i]$ on each axis and an admissible box $\mathcal{A}(x_t)$. Wrist force is the safety metric; the other intervals support its enforcement. (c) When that box is nonempty, componentwise clipping maps the blue proposal to the amber executed action. The coordinates are illustrative; the empty-intersection fallback is described in Section 4.1.

4.2 Planning with a Depth-Aware World Model

World model. CAP-WM plans with the latent model and MPPI planner of Section 3.3, whose training objective and planning procedure we keep (Appendix C.2). The force limit requires three changes: an observation that adds wrist depth and a history of controller readings, the action that training credits (Section 4.3), and the projection through which every executed action passes.

Observations. The observation $o_t = (o_t^{\text{prop}}, I_t)$ stacks the last h controller readings (pose and velocities relative to the fixed part, wrist force, previous smoothed command) with the actions that produced them, $h = 3$ on peg insertion and $h = 32$ on gear meshing and nut threading, so that the memoryless encoder can identify the per-episode smoothing factor and gains, and adds a 96×96 depth image I_t from a wrist-mounted camera. The modalities are encoded separately, summed and fused, $z_t = h_{\text{fuse}}\big(h_{\text{prop}}(o_t^{\text{prop}}) + h_{\text{depth}}(I_t)\big)$, following Hansen et al. (2026), with h_{depth} a four-layer strided convolutional network. The image supplies what proprioception and force cannot: the lateral offset between part and socket, which decides an insertion at sub-millimetre clearance, is invisible to the arm's own state and visible in the depth frame.

Planning. The first action of the sequence selected by (3), with exploration noise during training, is the proposal a_t; the executed action is $\Pi_{\mathcal{A}(x_t)}(a_t)$. The planner needs no access to the constraint: the layer reads x_t after the plan is made, immediately before the command is sent.

4.3 Training with the Projection in the Loop

Which action is credited. For a planner the variable the learner optimizes is the proposal: MPPI scores candidates by rolling the model out on them, the prior ascends $Q_\phi(z, \pi_\phi(z))$ at its own output, and TD targets are evaluated there. CAP-WM therefore stores the proposal a_t as the action of a transition, with the executed action beside it, so the model learns the dynamics of the composed system, environment together with projection, which is what the planner controls. Every action is projected, including the random actions of the seed phase, so replay never holds an unprojected transition; only the observation stack receives the executed action, since it recovers the smoothing from the command that produced the next reading (Appendix B.5).

Model-free instantiation. CAP-PPO wraps the same layer around a Gaussian policy π_θ with the recurrent architecture of Noseworthy et al. (2025), trained with PPO (Schulman et al., 2017). The raw sample $a_t \sim \pi_\theta(\cdot \mid o_t)$ is projected before execution and the importance ratio is taken on a_t, the action whose density the policy defines: from the policy's point of view the projection is part of the environment, and the estimator is unchanged. No gradient passes through the layer; the policy

adapts to the constraint through the advantages it earns in the projected system (Krasowski et al., 2023). The policy is trained from scratch with the projection active from the first iteration.

5 EXPERIMENTS

We evaluate whether CAP$^{\bullet}$ preserves task success under a pointwise force limit, how it compares with constrained RL and safety layers, and how performance depends on the learning backbone. We use the same measured-force criterion for all methods.

5.1 EXPERIMENTAL SETUP

Tasks and constraints. We evaluate CAP$^{\bullet}$ on three representative environments: peg insertion, gear meshing and nut threading in IsaacLab FORGE (Noseworthy et al., 2025). The tasks require aligning and assembling parts under uncertain relative pose and randomized impedance dynamics (Section 3.2). We retain FORGE's task rewards and success criteria. We choose force limits $F_{\max} \in \{5, 10\}$ N to test the tightest and most permissive endpoints of FORGE's force-penalty threshold range. Force compliance is evaluated on the wrist-force norm $\|F_t\|_2$ of Eq. (1) at every control step. The world-model agents additionally observe a wrist-mounted depth camera.

Baselines. We compare CAP$^{\bullet}$ with unconstrained learners, constrained RL and safety layers on the same tasks and force limits. PPO (FORGE) uses FORGE's recurrent PPO training recipe; TD-MPC2 + depth uses the visual world-model planner without projection. Safe RL comprises CPO, PID-Lagrangian and FOCOPS (Achiam et al., 2017; Stooke et al., 2020; Zhang et al., 2020), trained with accumulated excess force as the cost and a budget calibrated at each force limit from an unconstrained reference. Safety layer comprises SC^3 (Wang et al., 2026) and FCSCC (Liang et al., 2023), which filter the PPO (FORGE) policy of each task at execution time; we evaluate each filter with several settings of its contact-stiffness prior (Appendix C.3).

Training and selection. Model-free agents are trained with 128 parallel environments and world-model agents with 32. We select unconstrained checkpoints by success and constrained baselines by safe success; for CAP$^{\bullet}$, the most successful checkpoint without a violating episode, which in every cell is also the most successful overall (Appendix C.4). Each family row names the member selected at each cap. Training budgets, cost budgets and constraint parameters are listed in Appendix C.

Evaluation and metrics. Each row is evaluated on 640 episodes. Success rate (SR) is the fraction successful at the final step; safe success rate (SSR) additionally requires $\max_t \|F_t\|_2 \leq F_{\max}$. Excess force (EF) is the episode mean of $\sum_t [\|F_t\|_2 - F_{\max}]_+ \Delta t$, in N·s; peak force (PF) is the mean of the per-episode maxima, in N. We score these quantities from recorded episodes and force traces. A rate has binomial standard error at most 2.0 percentage points. We report success and force compliance together and do not interpret small success differences as a resolved ranking.

5.2 RESULTS

Task success under a force limit. CAP$^{\bullet}$ attains high safe success on every task, and none of its 7,680 evaluation episodes exceeds the limit: the result holds for each of the twelve CAP$^{\bullet}$ controllers, each trained separately and from scratch (Table 2). Pooled over these episodes, the average per-episode violation probability is below 0.04% at one-sided 95% confidence, and the largest episode peak is 4.78 N at the 5 N limit and 8.26 N at the 10 N limit. At the tighter limit, CAP-WM$^{\bullet}$ achieves 98.1% safe success on peg insertion, compared with 99.1% success and 0.6% safe success for TD-MPC2 + depth. Mean peak force falls from 10.60 to 1.56 N; Figures 3 and 7 show the episode-peak distributions. On gear meshing, CAP-WM$^{\bullet}$ retains 94.8% success compared with 95.2% for the unprojected planner, whose successful episodes all exceed the limit. On nut threading at 5 N, CAP-WM$^{\bullet}$ achieves 89.2% safe success versus 0% for TD-MPC2 + depth, while retaining 89.2% versus 96.2% task success.

Comparison with constrained baselines. Safe RL improves safe success over unconstrained learning on peg insertion and gear meshing, but remains below CAP$^{\bullet}$. At the tighter cap, the best Safe RL members attain 28.9% and 16.2% safe success, respectively. Relaxing the cap improves their performance, but does not remove violations. On nut threading at the tighter cap, FOCOPS has mean excess force 26.608 N·s, below its configured budget of 39.226 N·s, yet every episode violates

Table 2: **Success and force compliance** on the FORGE assembly tasks, 640 evaluation episodes per result. SR, SSR, EF and PF are defined in Section 5.1. Safe RL and Safety layer show the member with the highest SSR at each cap (lower/higher cap when different). **Bold**: highest SSR and lowest EF.

		Force limit $F_{\max} = 5\,\mathrm{N}$				Force limit $F_{\max} = 10\,\mathrm{N}$			
	Method	**SR** ↑ (%)	**SSR** ↑ (%)	**EF** ↓ (N·s)	**PF** ↓ (N)	**SR** ↑ (%)	**SSR** ↑ (%)	**EF** ↓ (N·s)	**PF** ↓ (N)
Peg insertion	PPO (FORGE)	97.7	0.3	2.424	10.26	97.7	49.5	0.122	10.26
	Safe RL (FOCOPS)	42.7	28.9	0.057	4.66	71.4	69.1	0.002	5.86
	Safety layer (FCSCC / SC^3)	90.2	6.1	0.243	7.01	91.9	58.3	0.559	9.58
	TD-MPC2 + depth	99.1	0.6	7.784	10.60	99.1	44.7	0.284	10.60
	CAP-PPO (ours)	78.0	78.0	**0.000**	1.75	90.5	90.5	**0.000**	3.45
	CAP-WM (ours)	98.1	**98.1**	**0.000**	**1.56**	98.0	**98.0**	**0.000**	3.15
Gear meshing	PPO (FORGE)	99.8	0.0	12.957	15.10	99.8	11.6	2.274	15.10
	Safe RL (PID-Lag. / FOCOPS)	31.6	16.2	1.704	4.96	75.8	58.3	0.039	8.53
	Safety layer (FCSCC / SC^3)	21.7	2.2	0.632	5.01	87.3	52.7	0.342	10.14
	TD-MPC2 + depth	95.2	0.0	20.253	13.48	95.2	11.6	1.849	13.48
	CAP-PPO (ours)	94.7	94.7	**0.000**	**1.89**	98.4	**98.4**	**0.000**	3.05
	CAP-WM (ours)	94.8	**94.8**	**0.000**	1.96	95.0	95.0	**0.000**	**3.00**
Nut threading	PPO (FORGE)	97.5	0.0	36.907	12.45	97.5	18.6	5.465	12.45
	Safe RL (FOCOPS)	90.2	0.0	26.608	9.40	90.2	64.8	0.039	9.40
	Safety layer (FCSCC)	45.0	1.1	0.787	6.66	77.5	72.0	0.059	8.49
	TD-MPC2 + depth	96.2	0.0	43.143	12.72	96.2	17.7	3.907	12.72
	CAP-PPO (ours)	61.4	61.4	**0.000**	1.85	87.2	87.2	**0.000**	3.22
	CAP-WM (ours)	89.2	**89.2**	**0.000**	**1.76**	91.9	**91.9**	**0.000**	**3.18**

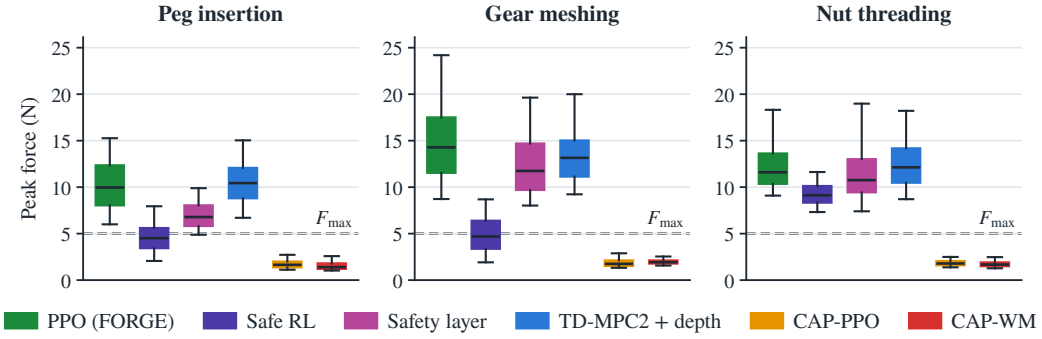

Figure 3: **Episode peak wrist force at the 5 N limit.** Each box summarizes the maximum control-step force norm in 640 evaluation episodes for one method and task. The line inside a box is the median, the box spans the 25th-75th percentiles, and whiskers mark the 5th-95th percentiles; values beyond the whiskers are not plotted. The dashed line marks $F_{\max}$.

the cap. This illustrates the distinction between satisfying a cumulative-cost budget and pointwise compliance. Both safety layers filter the same PPO (FORGE) policy as the first row, so the difference between the two rows isolates the filter. FCSCC trades success for compliance through its stiffness prior: on gear meshing at 10 N, at the identified stiffness it changes little (563 violating episodes against 566 for the unfiltered policy), and its stiffest setting lowers violations to 94 but success from 98.3% to 44.5%, whereas CAP-PPO reaches 98.4% with none. Even the setting with the highest safe success exceeds the limit in 71 to 602 of the 640 episodes, depending on task and limit (Appendix D.6).

Backbone and force-cap dependence. Both backbones achieve force compliance, while their task performance differs. CAP-WM outperforms CAP-PPO on peg insertion at the tighter cap; on gear meshing their safe success is comparable. At the looser cap, the gear ordering reverses: CAP-PPO reaches 98.4% safe success and CAP-WM reaches 95.0%. Overall, the world-model

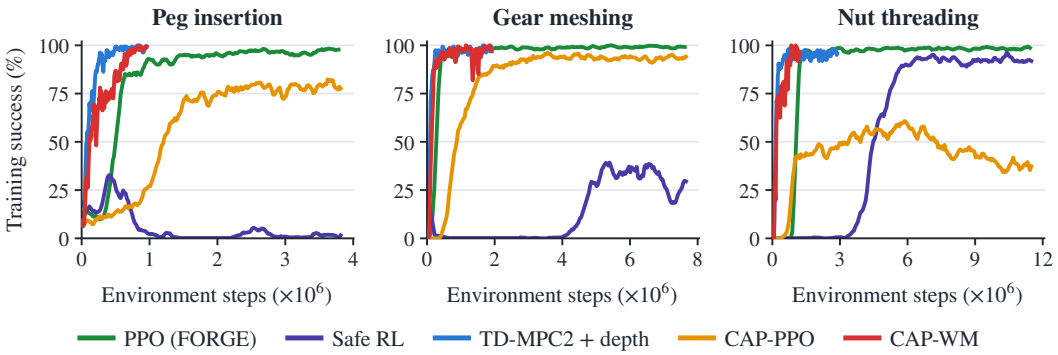

Figure 4: **Training success at the 5 N force limit.** Each curve is batch success during training, smoothed with a causal moving average over approximately 2.5% of logged points (at least three). The horizontal axis counts environment interactions; world-model runs use one quarter of the model-free step budget. Safety layers filter frozen policies and have no training curve. Curves measure success rate; Table 2 gives the held-out evaluation.

backbone achieves higher safe success on peg insertion and nut threading at both force limits. Backbone comparisons also include the different observations, training recipes and constraint parameters (Appendix C). Figures 4 and 6 show the corresponding learning behavior; high training success alone does not imply safe execution. Raising the cap improves safe success for the unconstrained agents, but their success/safe-success gaps persist on every task.

Projection activity and ablations. The reduction in force comes with frequent intervention rather than with a policy that rarely meets its constraint: across the twelve CAP$^{\blacklozenge}$ cells, the projection changes the translational action on 94.6-99.4% of control steps (Table 8 in Appendix D). The action range decides on at most 6.7% of steps and target-clip infeasibility never occurs, so by Corollary A.3 the bound on the applied wrench holds on every other step. Fixed-checkpoint ablations (Table 9) show that the projection is part of the learned behavior: removing it from a trained CAP$^{\blacklozenge}$ controller leaves at most 0.9% safe success on every task and at both limits, and task success falls as well. An evaluation-time force gate, which replaces the command by one predicted to apply zero force once the measured force exceeds $F_{\max} - \varepsilon$, lowers the excess force of the unconstrained agents at 5 N, yet at least 632 of 640 episodes still violate the limit, because the gate reacts to contact that has already occurred. A longer planning horizon at inference changes the success of CAP-WM$^{\blacklozenge}$ (50.6-96.2%) but hardly its compliance: one of 3,840 episodes exceeds the limit, by 0.13 N for a single control step (Table 10). Finally, on peg insertion, wrist depth raises success from 85.2% to 98.1% with the projection and from 89.7% to 99.1% without it, while only the projected models avoid violations (Table 11): depth supports task completion, and the projection provides compliance.

6 Conclusion

We introduced CAP$^{\blacklozenge}$, a closed-form action projection that enforces a per-step force limit on learned policies and world-model planners. CAP$^{\blacklozenge}$ derives admissible action intervals from the live state of the impedance controller, including its gains, action smoothing and dead zone, and clips each proposed action to them, which bounds the applied wrench at every step without an online solver or a learned constraint model. We combined it with a latent world model over wrist depth, proprioception and force, and trained CAP-WM$^{\blacklozenge}$ and CAP-PPO$^{\blacklozenge}$ with the projection in the loop. On three assembly tasks at two force limits, CAP$^{\blacklozenge}$ attains the highest safe success in every setting with no violation in any evaluation episode, whereas constrained RL and safety-layer baselines lose task success or frequently exceed the limit. CAP$^{\blacklozenge}$ has two main limitations. First, all experiments are in simulation, and transfer to physical hardware remains to be evaluated. Second, the projection relies on an explicit controller model: the bound holds for the applied wrench whenever the admissible set is nonempty, its extension to the measured force depends on a margin calibrated for each task, and another controller requires its own action-to-wrench map. Future work could validate CAP$^{\blacklozenge}$ on physical robots and extend it to controllers without a closed-form action-to-wrench map.

AI USE STATEMENT

We used large language models as writing assistants to improve the clarity and phrasing of the text, and as search aids when surveying related work. The research questions, the method, and the theoretical results were developed by the authors without LLM involvement, and all content was checked.

REPRODUCIBILITY STATEMENT

Code for training and evaluating CAP🛡 and the baselines, together with configuration files and model checkpoints, is made available at cap-wm.github.io. We rely on the FORGE tasks in IsaacLab (Noseworthy et al., 2025) for simulation, which are publicly available. All theoretical results are stated with explicit assumptions, and the complete proof of Proposition 1 is included in Appendix A. The construction and calibration of the projection are detailed in Appendix B, whose Algorithm 1 gives the control loop, and the experimental setup, including the tasks (Appendix C.1), hyperparameters, training budgets, checkpoint selection and evaluation metrics (Appendix C.5), is described in Appendix C, with the compute used in Appendix C.6.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31. PMLR, 2017.

Ahmed Agha, Baris Kayalibay, Atanas Mirchev, Patrick van der Smagt, and Justin Bayer. Exploring under constraints with model-based actor-critic and safety filters. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=s31IWg2kN5.

Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Eitan Altman. *Constrained Markov decision processes*. Chapman and Hall/CRC, 1999.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

Alexander W Goodall and Francesco Belardinelli. Leveraging approximate model-based shielding for probabilistic safety guarantees in continuous environments. *arXiv preprint arXiv:2402.00816*, 2024.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Oxh5CstDJU.

Nicklas Hansen, Iretiayo Akinola, Yijie Guo, Jie Xu, Bingjie Tang, Hao Su, Xiaolong Wang, Abhishek Gupta, Dieter Fox, and Yashraj Narang. Generalizable robotic insertion with world models. *arXiv preprint arXiv:2609.28258*, 2026.

Weidong Huang, Jiaming Ji, Borong Zhang, Chunhe Xia, and Yaodong Yang. SafeDreamer: Safe reinforcement learning with world models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=tsE5HLYtYg.

Hanna Krasowski, Jakob Thumm, Marlon Müller, Lukas Schäfer, Xiao Wang, and Matthias Althoff. Provably safe reinforcement learning: Conceptual analysis, survey, and benchmarking. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=mcN0ezbnzO.

Haiyan Liang, Xinming Wang, Jianliang Mao, and Jun Yang. Control barrier function-based force constrained safety compliance control for manipulator. In *2023 28th International Conference on Automation and Computing (ICAC)*, pp. 1–7, 2023. doi: 10.1109/ICAC57885.2023.10275266.

Puze Liu, Davide Tateo, Haitham Bou Ammar, and Jan Peters. Robot reinforcement learning on the constraint manifold. In *5th Annual Conference on Robot Learning*, 2021. URL https://openreview.net/forum?id=zwo1-MdMl1P.

Rishabh Madan, Angchen Xie, Samantha Saak, Andres Blanco, Dohyeok Lee, Sarah Grace Brown, Yunting Yan, Mark Zolotas, Jose Barreiros, and Tapomayukh Bhattacharjee. TACTIC: Tactile and vision conditioned contact-centric control for whole-arm manipulation. In *Robotics: Science and Systems (RSS)*, 2026.

Hannah Markgraf, Shambhuraj Sawant, Hanna Krasowski, Lukas Schäfer, Sebastien Gros, and Matthias Althoff. Safe reinforcement learning using action projection: Safeguard the policy or the environment? *arXiv preprint arXiv:2509.12833*, 2025.

Janine Matschek, Johanna Bethge, and Rolf Findeisen. Safe machine-learning-supported model predictive force and motion control in robotics. *IEEE Transactions on Control Systems Technology*, 31(6):2380–2392, 2023.

Youngjae Min and Navid Azizan. HardNet: Hard-constrained neural networks with universal approximation guarantees. *arXiv preprint arXiv:2410.10807*, 2024.

Yashraj Narang, Kier Storey, Iretiayo Akinola, Miles Macklin, Philipp Reist, Lukasz Wawrzyniak, Yunrong Guo, Adam Moravanszky, Gavriel State, Michelle Lu, Ankur Handa, and Dieter Fox. Factory: Fast contact for robotic assembly. In *Robotics: Science and Systems*, 2022.

Michael Noseworthy, Bingjie Tang, Bowen Wen, Ankur Handa, Chad Kessens, Nicholas Roy, Dieter Fox, Fabio Ramos, Yashraj Narang, and Iretiayo Akinola. FORGE: Force-guided exploration for robust contact-rich manipulation under uncertainty. *IEEE Robotics and Automation Letters*, 10 (5):4436–4443, 2025. doi: 10.1109/LRA.2025.3551637.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by PID Lagrangian methods. In *International conference on machine learning*, pp. 9133–9143. PMLR, 2020.

Jannick Stranghöner, Philipp Hartmann, Marco Braun, Sebastian Wrede, and Klaus Neumann. SHaRe-RL: Structured, interactive reinforcement learning for contact-rich industrial assembly tasks. *arXiv preprint arXiv:2509.13949*, 2025.

Petr Svarny, Michael Tesar, Jan Kristof Behrens, and Matej Hoffmann. Safe physical HRI: Toward a unified treatment of speed and separation monitoring together with power and force limiting. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7580–7587. IEEE, 2019.

Bingjie Tang, Michael A Lin, Iretiayo Akinola, Ankur Handa, Gaurav S Sukhatme, Fabio Ramos, Dieter Fox, and Yashraj Narang. IndustReal: Transferring contact-rich assembly tasks from simulation to reality. In *Robotics: Science and Systems*, 2023.

Bingjie Tang, Iretiayo Akinola, Jie Xu, Bowen Wen, Ankur Handa, Karl Van Wyk, Dieter Fox, Gaurav S Sukhatme, Fabio Ramos, and Yashraj Narang. AutoMate: Specialist and generalist assembly policies over diverse geometries. In *Robotics: Science and Systems*, 2024.

Xinming Wang, Jun Yang, Jianliang Mao, Shihua Li, and Yunda Yan. Guarding force: Safety-critical compliant control for robot-environment interaction. *IEEE Robotics and Automation Letters*, 11 (1):930–937, 2026. doi: 10.1109/LRA.2025.3636028.

Grady Williams, Andrew Aldrich, and Evangelos A Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40 (2):344–357, 2017.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. DayDreamer: World models for physical robot learning. In *Conference on Robot Learning*, pp. 2226–2240. PMLR, 2022.

Heng Zhang, Gokhan Solak, Gustavo JG Lahr, and Arash Ajoudani. SRL-VIC: A variable stiffness-based safe reinforcement learning for contact-rich robotic tasks. *IEEE Robotics and Automation Letters*, 9(6):5631–5638, 2024.

Heng Zhang, Gokhan Solak, Sebastian Hjorth, and Arash Ajoudani. Passivity-centric safe reinforcement learning for contact-rich robotic tasks. *Robotics and Autonomous Systems*, pp. 105663, 2026.

Yiming Zhang, Quan Vuong, and Keith Ross. First order constrained optimization in policy space. *Advances in Neural Information Processing Systems*, 33:15338–15349, 2020.

Weiye Zhao, Rui Chen, Yifan Sun, Feihan Li, Tianhao Wei, and Changliu Liu. State-wise constrained policy optimization. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=NgK5etmhz9.

Hao Zhou, Yanze Zhang, Cameron Reid, and Wenhao Luo. Certifiable safe model-based reinforcement learning with control-affine dynamics approximation. *arXiv preprint arXiv:2607.16501*, 2026.

A PROOFS

This section proves Proposition 1 and describes what holds on the steps that the proposition does not cover.

A.1 NOTATION AND ASSUMPTIONS

Fix a decision instant t. All quantities below are read from the same controller state x_t, and we drop the dependence on x_t where it is unambiguous; Table 3 lists the symbols. For each translational axis $i \in \{1, 2, 3\}$ we assume

$$K_{p,i} > 0, \qquad B_i > 0, \qquad \lambda > 0, \qquad d_i \geq 0, \qquad \delta_i \geq 0, \qquad \Lambda_i > 0, \qquad \bar{v}_{t,i} \geq 0, \\ \kappa_i > 0, \qquad \textstyle\sum_{j=1}^{3} \kappa_j^2 \leq 1, \qquad 0 \leq \bar{F}_t \leq F_{\max} - \varepsilon. \tag{9}$$

Each condition holds by construction: the action scale is a positive constant, the gains, the smoothing factor and the clip are drawn from positive ranges (Appendix C.1), the speed budget is at least the touchdown speed (Appendix B.3), $\kappa_i = 1/\sqrt{3}$, and the force budget is clamped to $[0, F_{\max} - \varepsilon]$ (Appendix B.2). The first three conditions give $m_i = K_{p,i} B_i \lambda > 0$, so the map $a_i \mapsto c_i + m_i a_i$ of (4) is strictly increasing. Λ_i is the inertia used by the speed interval (Appendix B.1). We write $\eta_i = K_{p,i} |v_{t,i}| \Delta t$ for the drift allowance of (5) and $b_i = d_i + (\kappa_i \bar{F}_t - \eta_i)_+$ for the half-width of the force interval, so that $\mathcal{I}_i^{\text{force}} = \{a_i : |c_i + m_i a_i| \leq b_i\}$. All statements concern the translational wrench at the decision instant and the one-step free-body velocity model of Section 4.1.

Table 3: **Notation.** Vectors have three translational components unless stated otherwise. Controller parameters and state are evaluated at the current decision instant.

Symbol	Meaning
$a_t,\ a_t^{\text{exec}}$	Proposed and executed actions in $[-1, 1]^7$
$\tilde{a}_{t-1},\ \lambda$	Previous smoothed action and action-smoothing factor
$p_t,\ v_t,\ p^{\text{ref}}$	Fingertip position, fingertip velocity and position reference
$K_p,\ K_d,\ B,\ \delta$	Stiffness, damping, action scale and position-error clip (diagonal)
$c_i,\ m_i$	Intercept and slope of the commanded wrench, $W_{t,i} = c_i + m_i a_i$
$d_i,\ D_{d_i}$	Dead-zone width and dead-zone map, $W_{t,i}^{\text{app}} = D_{d_i}(W_{t,i})$
$F_t,\ F_{\max},\ \varepsilon$	Wrist force, force limit and calibrated margin
$\bar{F}_t,\ \kappa_i$	Force budget of the step and its share on axis i
$\eta_i,\ b_i$	Drift allowance $K_{p,i}\lvert v_{t,i}\rvert\Delta t$ and half-width of the force interval
$\Lambda_i,\ \bar{v}_{t,i}$	Inertia used by the speed interval and speed budget of axis i
$\ell_i,\ u_i$	Bounds on the applied wrench imposed by the speed budget
$L_i,\ U_i$	Endpoints of the admissible action interval of axis i
Δt	Control period, $1/15$ s

A.2 AUXILIARY LEMMAS

The proposition rests on two facts: the dead zone maps intervals of applied wrench back to intervals of commanded wrench, and clipping through several intervals in sequence equals clipping onto their intersection.

Lemma A.1 (Dead zone)**.** *Let $d \geq 0$ and $D_d(w) = \operatorname{sign}(w)\,(|w| - d)_+$.* (i) *$D_d$ is continuous, nondecreasing and 1-Lipschitz, and $|D_d(w)| = (|w| - d)_+$.* (ii) *For $\ell \leq u$, define*

$$g_d^-(\ell) = \begin{cases} \ell - d, & \ell \leq 0, \\ \ell + d, & \ell > 0, \end{cases} \qquad g_d^+(u) = \begin{cases} u - d, & u < 0, \\ u + d, & u \geq 0. \end{cases} \tag{10}$$

Then

$$\{w : \ell \leq D_d(w) \leq u\} = \big[g_d^-(\ell),\, g_d^+(u)\big], \tag{11}$$

and this interval is nonempty. (iii) *For $m > 0$ and $c \in \mathbb{R}$, $\{a : \ell \leq D_d(c + ma) \leq u\} = \big[(g_d^-(\ell) - c)/m,\, (g_d^+(u) - c)/m\big]$.*

Proof. On its three branches,

$$D_d(w) = \begin{cases} w + d, & w < -d, \\ 0, & -d \le w \le d, \\ w - d, & w > d, \end{cases} \tag{12}$$

the map is affine with slopes 1, 0 and 1 and is continuous at $w = \pm d$, which gives (i). For (ii), consider the lower bound first. If $\ell > 0$, then $D_d(w) \ge \ell$ holds only on the positive branch, where it reads $w \ge \ell + d$. If $\ell \le 0$, it holds on the zero and positive branches and, on the negative branch, exactly when $w \ge \ell - d$; since $\ell - d \le -d$, the set is $\{w \ge \ell - d\}$. In both cases the lower bound is $w \ge g_d^-(\ell)$. The same argument turns $D_d(w) \le u$ into $w \le u - d$ if $u < 0$ and into $w \le u + d$ if $u \ge 0$, that is, $w \le g_d^+(u)$. The interval is nonempty: if $\ell \le 0 \le u$, then $g_d^-(\ell) \le -d \le d \le g_d^+(u)$; if $u < 0$, then $g_d^-(\ell) = \ell - d \le u - d$; and if $\ell > 0$, then $\ell + d \le u + d$. Part (iii) follows from (ii) because $a \mapsto c + ma$ is strictly increasing for $m > 0$. □

For $\ell = u = 0$, (11) is the whole dead band $[-d, d]$, endpoints included. The force interval is the case $u_i = -\ell_i = (\kappa_i \bar{F}_t - \eta_i)_+$ of Lemma A.1(iii), for which $g_{d_i}^-(\ell_i) = -b_i$ and $g_{d_i}^+(u_i) = b_i$:

$$\mathcal{I}_i^{\text{force}} = \left\{ a_i : \ |D_{d_i}(c_i + m_i a_i)| \le (\kappa_i \bar{F}_t - \eta_i)_+ \right\}. \tag{13}$$

Applying Lemma A.1(iii) to the three constraints of Section 4.1 gives their endpoints in closed form:

$$L_i^{\text{force}} = \frac{-b_i - c_i}{m_i}, \qquad U_i^{\text{force}} = \frac{b_i - c_i}{m_i}, \tag{14}$$

$$L_i^{\text{vel}} = \frac{g_{d_i}^-(\ell_i) - c_i}{m_i}, \qquad U_i^{\text{vel}} = \frac{g_{d_i}^+(u_i) - c_i}{m_i}, \tag{15}$$

$$L_i^{\text{clip}} = \frac{-K_{p,i}\delta_i - c_i - K_{d,i} v_{t,i}}{m_i}, \qquad U_i^{\text{clip}} = \frac{K_{p,i}\delta_i - c_i - K_{d,i} v_{t,i}}{m_i}, \tag{16}$$

with $\ell_i = -\Lambda_i(\bar{v}_{t,i} + v_{t,i})/\Delta t$ and $u_i = \Lambda_i(\bar{v}_{t,i} - v_{t,i})/\Delta t$ as in (6). Since $u_i - \ell_i = 2\Lambda_i \bar{v}_{t,i}/\Delta t \ge 0$, the speed interval is nonempty. When $v_{t,i} > \bar{v}_{t,i}$, both bounds on the applied wrench are negative and the interval demands braking; the sign cases of (10) preserve this demand. The admissible interval of axis i is $[L_i, U_i]$ with

$$L_i = \max\{-1, L_i^{\text{clip}}, L_i^{\text{vel}}, L_i^{\text{force}}\}, \qquad U_i = \min\{1, U_i^{\text{clip}}, U_i^{\text{vel}}, U_i^{\text{force}}\}, \tag{17}$$

and the translational part of $\mathcal{A}(x_t)$ is nonempty if and only if $L_i \le U_i$ for $i = 1, 2, 3$. Equality is allowed and fixes the action of that axis.

Lemma A.2 (Sequential clipping)**.** *Let $I_j = [l_j, u_j]$, $j = 1, \dots, k$, be closed intervals with nonempty intersection $I = [L, U]$, and let $C_I(q) = \min\{U, \max\{L, q\}\}$. Then*

$$C_{I_k} \circ \cdots \circ C_{I_1}(q) = C_I(q) \qquad \textit{for every } q \in \mathbb{R}. \tag{18}$$

Consequently, clipping each translational coordinate through the action range, the target-clip, speed and force intervals, and the action range again yields the Euclidean projection (8) whenever $\mathcal{A}(x_t) \neq \emptyset$.

Proof. Let I_1 and I_2 intersect and write $L = \max\{l_1, l_2\} \le U = \min\{u_1, u_2\}$. If $q \in [L, U]$, neither clip moves q. Let $q < L$. If $L = l_2$, then $C_{I_1}(q) \le C_{I_1}(L) = L$ by monotonicity and $L \in I_1$, so the second clip returns L. If $L = l_1 > l_2$, then $C_{I_1}(q) = L \in I_2$, and the second clip leaves it. The case $q > U$ is symmetric. Hence $C_{I_2} \circ C_{I_1} = C_{I_1 \cap I_2}$. Every partial intersection $I_1 \cap \cdots \cap I_j$ contains I and is therefore nonempty, so induction on k gives (18). For the projection, $\mathcal{A}(x_t)$ is a product of intervals and $\|y - a\|_2^2 = \sum_i (y_i - a_i)^2$ separates over coordinates; on $[L_i, U_i]$ the unique minimizer of $(y_i - a_i)^2$ is $C_{[L_i, U_i]}(a_i)$. Repeating the action range does not change the intersection. □

A.3 Proof of Proposition 1

Proposition 1 (Per-step force bound). *Let $\mathcal{A}(x_t) \neq \emptyset$. For every proposed action $a_t \in \mathcal{A}$, the executed action $a_t^{\mathrm{exec}} = \Pi_{\mathcal{A}(x_t)}(a_t)$ leaves the target clip inactive, so that the commanded wrench is given by (4); the applied wrench satisfies $|W_{t,i}^{\mathrm{app}}| \leq \kappa_i \bar{F}_t$ for $i = 1, 2, 3$ and hence $\|W_t^{\mathrm{app}}\|_2 \leq \bar{F}_t \leq F_{\max} - \varepsilon$; and the free-body velocity after the step satisfies $|v_{t,i} + \Delta t\, W_{t,i}^{\mathrm{app}}/\Lambda_i| \leq \bar{v}_{t,i}$.*

Proof. Since $\mathcal{A}(x_t) \neq \emptyset$, the executed action lies in every interval of (7); for the sequential clip of the implementation this is Lemma A.2. Let $\tilde{a}_{t,i} = \lambda a_{t,i}^{\mathrm{exec}} + (1-\lambda)\tilde{a}_{t-1,i}$ be the smoothed action. By the definition of c_i and m_i below (4),

$$c_i + K_{d,i} v_{t,i} + m_i a_{t,i}^{\mathrm{exec}} = K_{p,i}\big(p_i^{\mathrm{ref}} + B_i \tilde{a}_{t,i} - p_{t,i}\big) = K_{p,i}\big(p_{t,i}^{\mathrm{d}} - p_{t,i}\big). \tag{19}$$

Membership in $\mathcal{I}_i^{\mathrm{clip}}$ bounds the left side by $K_{p,i}\delta_i$ in absolute value, and dividing by $K_{p,i} > 0$ gives $|p_{t,i}^{\mathrm{d}} - p_{t,i}| \leq \delta_i$. On this range the clip in (2) is the identity, endpoints included, so the commanded wrench is exactly $W_{t,i} = c_i + m_i a_{t,i}^{\mathrm{exec}}$.

Membership in $\mathcal{I}_i^{\mathrm{force}}$ gives $|W_{t,i}| \leq b_i$, and Lemma A.1(i) gives

$$|W_{t,i}^{\mathrm{app}}| = (|W_{t,i}| - d_i)_+ \leq (b_i - d_i)_+ = (\kappa_i \bar{F}_t - \eta_i)_+ \leq \kappa_i \bar{F}_t, \tag{20}$$

where the last step uses $\eta_i \geq 0$ and $\kappa_i \bar{F}_t \geq 0$. Summing squares, $\|W_t^{\mathrm{app}}\|_2^2 \leq \bar{F}_t^2 \sum_i \kappa_i^2 \leq \bar{F}_t^2$, and $\bar{F}_t \leq F_{\max} - \varepsilon$ by (9).

Finally, membership in $\mathcal{I}_i^{\mathrm{vel}}$ and Lemma A.1(iii) give $\ell_i \leq W_{t,i}^{\mathrm{app}} \leq u_i$. Multiplying by $\Delta t/\Lambda_i > 0$ and adding $v_{t,i}$,

$$-\bar{v}_{t,i} \leq v_{t,i} + \frac{\Delta t}{\Lambda_i} W_{t,i}^{\mathrm{app}} \leq \bar{v}_{t,i}, \tag{21}$$

which is the velocity bound. □

The same argument applies when the dead-zone guard of Appendix B.2 replaces b_i by a smaller half-width $b_i^{\mathrm{guard}} \leq b_i$, since $|W_{t,i}| \leq b_i^{\mathrm{guard}}$ implies $|W_{t,i}| \leq b_i$. For a zero budget, $\bar{F}_t = 0$, each force interval requires $|W_{t,i}| \leq d_i$ and the modeled applied wrench is zero; the construction never divides by the budget.

A.4 Remarks on the Assumptions

Proposition 1 assumes a nonempty admissible set and bounds the modeled applied wrench at the decision instant. We now consider steps whose intervals do not intersect, the relation to the measured force, the motion within a control period, and the inertia used by the speed interval.

Empty intersections. Each interval has ordered endpoints, by $K_{p,i}\delta_i \geq 0$ for the target-clip interval, Lemma A.1(ii) for the speed interval and $b_i \geq 0$ for the force interval, so only their intersection can be empty. The implementation always returns an action through the ordered clips of Lemma A.2: target clip, speed, force, and then the action range. When the intervals of an axis do not intersect, the force interval therefore takes precedence over the speed and target-clip intervals, and the action range takes precedence over all three.

Corollary A.3 (Force bound on unsaturated steps). *Let x_t satisfy (9), and let a_t^{exec} be the output of the ordered clips, whether or not $\mathcal{A}(x_t)$ is empty. If the final clip to the action range does not change the translational coordinates, and $a_{t,i}^{\mathrm{exec}} \in \mathcal{I}_i^{\mathrm{clip}}(x_t)$ for $i = 1, 2, 3$, then $|W_{t,i}^{\mathrm{app}}| \leq \kappa_i \bar{F}_t$ for $i = 1, 2, 3$ and $\|W_t^{\mathrm{app}}\|_2 \leq \bar{F}_t \leq F_{\max} - \varepsilon$.*

Proof. Each force interval is nonempty because $b_i \geq 0$, so the output of the force clip lies in $\mathcal{I}_i^{\mathrm{force}}$. By hypothesis the final clip leaves it unchanged, so $a_{t,i}^{\mathrm{exec}} \in \mathcal{I}_i^{\mathrm{force}} \cap \mathcal{I}_i^{\mathrm{clip}}$. The first two parts of the proof of Proposition 1 use only these two memberships. □

The two hypotheses exclude exactly the steps that the evaluation flags as saturated or as target-clip infeasible (Appendix C.5). A step whose braking demand exceeds the force budget still satisfies the

force bound, which is the force priority of Section 4.1. Target-clip infeasibility does not occur in any evaluation of Table 2 (Table 8), so in these evaluations the bound on the modeled applied wrench holds on every unsaturated step.

Measured force. The limit in (1) applies to the wrist force read at the decision instants. We index the reading by the step it follows: F_t is read at the end of the period in which a_t^{exec} acts, and F_{t-1}, the reading available when a_t^{exec} is chosen, sets the budget $\bar{F}_t$. On every step covered by Proposition 1 or Corollary A.3, a measured force that exceeds the modeled applied wrench by at most the margin satisfies the limit:

$$\|F_t\|_2 \le \|W_t^{\mathrm{app}}\|_2 + \varepsilon \le \bar{F}_t + \varepsilon \le F_{\max}. \tag{22}$$

The first inequality is a property of the contact rather than of the controller model, and the margin is calibrated to it: Appendix B.3 estimates ε from the positive part of $\|F_t\|_2 - \|W_t^{\mathrm{app}}\|_2$ on recorded traces. Impact transients, which this quasi-static relation does not describe, are the object of the speed interval. Table 2 and Figure 7 evaluate the measured force directly.

Motion within the control period. Proposition 1 concerns the wrench at the decision instant, while the controller evaluates (2) at 120 Hz against a moving arm with the target held. At the current velocity the proportional term changes by at most η_i over one period, and the force interval reserves this amount. If the commanded wrench within the period is $W_{t,i} + e_i$ with $|e_i| \le \eta_i \le \kappa_i \bar{F}_t$ and the dead zone does not change, Lemma A.1(i) and (20) give

$$|D_{d_i}(W_{t,i} + e_i)| \le |D_{d_i}(W_{t,i})| + |e_i| \le (\kappa_i \bar{F}_t - \eta_i) + \eta_i = \kappa_i \bar{F}_t. \tag{23}$$

A change of the dead zone within the period is handled by the guard of Appendix B.2.

Scaled inertia. The speed interval uses $\Lambda_i = \rho\, \Lambda_i^{\mathrm{arm}}$, a scaled copy of the arm's task-space inertia Λ_i^{arm} with $0 < \rho \le 1$ (Appendix B.1). Let $v_i^+ = v_i + \Delta t\, W_i^{\mathrm{app}}/\Lambda_i$ be the velocity that the interval bounds. The free-body update with the arm's inertia is

$$v_i + \frac{\Delta t}{\Lambda_i^{\mathrm{arm}}} W_i^{\mathrm{app}} = (1-\rho)\, v_i + \rho\, v_i^+. \tag{24}$$

If v_i and v_i^+ both lie in $[-\bar{v}_i, \bar{v}_i]$, so does their convex combination. The scaling is therefore conservative for a state within its budget; for a state above it, the speed interval demands the recovery of the scaled model.

B Implementation Details

This section describes how the projection is computed and calibrated, the task-specific yaw constraints, and how the two backbones learn with the projection in the loop.

B.1 Controller State and Computational Cost

CAP$^{\blacklozenge}$ reads the controller state from the simulator's controller interface immediately before each environment step; it does not use the agent's observation. Besides the quantities of Section 3.2, namely the gains, the smoothing factor and filter memory, the dead-zone widths, the position-error clips and the fingertip state, the speed interval uses the height g_t of the held part above the fixed part and the arm's inertia, and the dead-zone guard uses the time until the next dead-zone redraw. From the translational Jacobian J and the joint-space mass matrix M_q we form the directional task-space inertia

$$\Lambda_i^{\mathrm{arm}} = \left[(J M_q^{-1} J^\top)_{ii}\right]^{-1}, \qquad \Lambda_i = \rho\, \Lambda_i^{\mathrm{arm}}, \qquad \rho = 0.8. \tag{25}$$

It is computed from the arm model at every step and is not learned. Algorithm 1 summarizes one control step of CAP-WM$^{\blacklozenge}$, including what is written to the replay buffer. CAP-PPO$^{\blacklozenge}$ executes the same projection; it stores the raw Gaussian sample for its likelihood ratio (Appendix B.5).

Computational cost. The interval arithmetic is batched over environments and uses a constant number of elementwise operations per axis. On one CPU thread (Intel Core i9-13900KF, PyTorch 2.7), one call of the layer on a batch of 128 environments, including its diagnostics, takes 0.34 to 0.39 ms (median of 3,000 calls), under 0.6% of the 66.7 ms control period. The inertia estimate (25) adds one 7×7 linear solve per environment. Over a full evaluation of 640 episodes on the same machine, CAP-WM$^{\blacklozenge}$ took 848 s against 858 s for TD-MPC2 + depth on peg insertion, and 1,126 s against 1,137 s on gear meshing.

Algorithm 1 One control step of CAP-WM$^{\blacklozenge}$ during training

Require: Observation history o_t, controller state x_t, wrist force F_{t-1}
1: Propose a_t: the first action of the MPPI solution of (3), with exploration noise
2: Compute c_i, m_i, η_i, the force budget (26), the guard (27) and the speed budget (28)
3: Form $\mathcal{I}_i^{\text{clip}}$, $\mathcal{I}_i^{\text{vel}}$, $\mathcal{I}_i^{\text{force}}$ from (14)–(16)
4: $q \leftarrow \text{clip}(a_t; [-1, 1]^7)$
5: **for** $i = 1, 2, 3$ **do**
6: $\quad q_i \leftarrow C_{\mathcal{I}_i^{\text{force}}}\big(C_{\mathcal{I}_i^{\text{vel}}}(C_{\mathcal{I}_i^{\text{clip}}}(q_i))\big)$
7: **end for**
8: $a_t^{\text{exec}} \leftarrow \text{clip}(q; [-1, 1]^7)$; apply the task's yaw constraint, if any (Appendix B.4)
9: Execute a_t^{exec}; observe r_t, o_{t+1} and F_t
10: Store (o_t, a_t, r_t, o_{t+1}) in the replay buffer, with a_t^{exec} beside it
11: Append a_t^{exec} and the new controller reading to the observation history

B.2 Force Budget and Dead-Zone Guard

With $\bar{F} = F_{\max} - \varepsilon$, the force budget of a step is

$$\bar{F}_t = \min\big\{\bar{F},\ \max\{0,\ \beta\bar{F} + (1-\beta)(F_{\max} - \|F_{t-1}\|_2)\}\big\}, \qquad \beta = 0.5, \tag{26}$$

where $\|F_{t-1}\|_2$ is the wrist force at the end of the previous period, read before observation noise. The first decision of an episode uses $\bar{F}$. If $\|F_{t-1}\|_2 \le \varepsilon$, the full budget is available; above that, the budget decreases linearly in the measured force and reaches zero at $\|F_{t-1}\|_2 = F_{\max} + \bar{F}$. In every case $0 \le \bar{F}_t \le \bar{F}$, as (9) requires.

FORGE redraws the dead-zone widths of all environments every 2 s, that is, every 30 decisions. The dead-zone credit d_i in b_i is valid only for the current width: a commanded wrench that a wide dead zone absorbs is transmitted almost in full once the width is redrawn small. We therefore bound the commanded wrench ahead of each redraw. Let $j_t \ge 0$ be the number of decisions until the last decision before the redraw ($j_t = 0$ at that decision). With rate ζ and slack s,

$$S_{t,i} = \kappa_i(\bar{F} - s)_+ + j_t\,\zeta\,m_i, \qquad b_i^{\text{guard}} = \min\big\{b_i,\ (S_{t,i} - \eta_i)_+\big\}, \tag{27}$$

and b_i^{guard} replaces b_i in (14). All reported configurations use $\zeta = 1$ and $s = 0$. Since $b_i^{\text{guard}} \le b_i$, the guard only tightens the force interval, and Proposition 1 and Corollary A.3 hold unchanged. At $j_t = 0$, $|W_{t,i}| \le \kappa_i\bar{F}$ whatever the current width, and since $|D_{d'}(w)| \le |w|$ for every $d' \ge 0$, the same wrench satisfies the per-axis bound after any redraw. The bound $S_{t,i}$ grows by ζm_i per decision of lead time, so the guard binds only in a short window before each redraw.

B.3 Calibration and the Speed Budget

The margin ε and the impact slope σ are calibrated per task, and on gear meshing also per force limit and backbone; the gap margin g_0 and the deceleration ceiling $a_{\max}$ are set once per task. All four come from traced 640-episode evaluations of separate calibration runs (the CAP-PPO$^{\blacklozenge}$ learner with the projection disabled, or an earlier CAP$^{\blacklozenge}$ controller on the same task), not from the evaluations reported in Table 2. Table 4 lists the values.

Margin. A trace row t records the modeled applied-wrench norm $w_t = \|D_d(c + m\,a_t^{\text{exec}})\|_2$, the fingertip speed s_t and the height g_t, all read at decision t, and the wrist-force norm f_t read at the end of the same period. Let $e_t = (f_t - w_t)_+$. Slow-contact rows satisfy $f_t \ge 0.35\,\text{N}$ and $s_t \le 0.02\,\text{m/s}$; free-space rows satisfy $f_t < 0.35\,\text{N}$, $g_t > 5\,\text{mm}$ and $w_t < 0.05\,\text{N}$. The margin is the 99th percentile of e_t over slow-contact rows plus the 99th percentile of f_t over free-space rows plus $0.005\,\text{N}$; for peg insertion, $0.646 + 0.133 + 0.005 = 0.784\,\text{N}$.

Impact slope and touchdown speed. A contact onset is a row at which f_t crosses $0.35\,\text{N}$ from below. Its peak is the larger force of the onset row and the following row, and its approach speed is the speed at the preceding decision. For onsets approached at 5 mm/s or faster, we divide the positive excess of the peak over the modeled applied wrench by the approach speed; σ is the 99th percentile of this ratio. The slope therefore describes the part of the force peak that the applied-wrench model

does not account for. The touchdown speed is $v_{\text{touch}} = F_{\text{imp}}/\sigma$, where the impact allowance F_{imp} is fixed at about 47% of $F_{\max} - \varepsilon$ in every configuration.

Gap margin and deceleration ceiling. The gap margin g_0 is the 99th percentile of the height reading at the first contact onset of each episode, clipped at zero and rounded up to the next 0.5 mm; it places the touchdown speed at the heights where first contact occurs. The braking curve assumes a constant deceleration, whereas the action filter limits how fast the commanded wrench can change. We therefore cap the deceleration at $a_{\max}$, estimated from pre-contact rows in which the speed exceeds its budget: the mean deceleration over such rows of an episode, at the 25th percentile over episodes, rounded down to a multiple of $0.005\,\text{m/s}^2$. This gives $a_{\max} = 0.02\,\text{m/s}^2$ on peg insertion; on gear meshing and nut threading we use the estimator's floor of $0.005\,\text{m/s}^2$. The braking capacity of the controller itself, $(K_{p,i}\delta_i - d_i)/(2\Lambda_i)$, exceeds $a_{\max}$ in every reported configuration, so $a_{\max}$ sets the curve.

Speed budget. With the effective gap $g_t^{\text{eff}} = (g_t - g_0 - \|v_t\|_2 \Delta t)_+$, the speed budget is

$$\begin{aligned} \bar{v}_t &= \min\left\{v_{\text{ceil}},\ \max\{v_{\text{touch}}, v_t^*\}\right\}, \\ v_t^* &= -a_{\max}\Delta t + \sqrt{(a_{\max}\Delta t)^2 + v_{\text{touch}}^2 + 2a_{\max} g_t^{\text{eff}}}\,, \end{aligned} \tag{28}$$

with $v_{\text{ceil}} = 0.25\,\text{m/s}$; the budget is the same on the three axes. The root v_t^* is the nonnegative solution of $v^2 = v_{\text{touch}}^2 + 2a_{\max}(g_t^{\text{eff}} - v\Delta t)$: a part approaching at the budget speed can still brake to v_{touch} after one further period of travel, and g_t^{eff} already discounts the distance $\|v_t\|_2\Delta t$ that the part covers before the next decision. Since $v_t^* \geq v_{\text{touch}}$ exactly when $g_t^{\text{eff}} \geq v_{\text{touch}}\Delta t$, the budget equals v_{touch} near the surface and grows with the gap beyond it (Figure 5b). The budget enters the interval through (15). For gear meshing at 5 N, CAP-WM$^{\blacklozenge}$ uses $v_{\text{touch}} = 14.6$ mm/s below $g_t = -6$ mm, where the held gear is already engaged, and F_{imp}/σ above; every other configuration uses a single touchdown speed.

Table 4: **Projection parameters.** Calibrated margin ε, enforced bound $F_{\max} - \varepsilon$ on the modeled applied wrench, impact allowance F_{imp}, impact slope σ, gap margin g_0 and deceleration ceiling $a_{\max}$ (Appendix B.3). Where the two backbones differ, the entry reads CAP-PPO$^{\blacklozenge}$ / CAP-WM$^{\blacklozenge}$.

Task	$F_{\max}$ (N)	ε (N)	$F_{\max} - \varepsilon$ (N)	F_{imp} (N)	σ (N·s/m)	g_0 (mm)	$a_{\max}$ (m/s^2)
Peg insertion	5	0.784	4.216	2.000	201.7	4.0	0.02
	10	0.784	9.216	4.372	201.7	4.0	0.02
Gear meshing	5	0.893 / 0.899	4.107 / 4.101	1.948	211.1 / 266.3	27.5	0.005
	10	1.138	8.862	4.204	266.3	27.5	0.005
Nut threading	5	0.680	4.320	2.049	251.1	27.0	0.005
	10	0.680	9.320	4.421	251.1	27.0	0.005

B.4 Task-Specific Yaw Constraints

The yaw command is the sixth action coordinate. Its smoothed target is

$$\psi^{\text{d}} = O - S\big[\lambda a_{t,6} + (1-\lambda)\tilde{a}_{t-1,6}\big], \qquad O = \pi/4, \quad S = 3\pi/4, \tag{29}$$

which is strictly decreasing in $a_{t,6}$. A target interval $[\psi_-, \psi_+]$ therefore has the action preimage

$$\left[\frac{(O-\psi_+)/S - (1-\lambda)\tilde{a}_{t-1,6}}{\lambda},\ \frac{(O-\psi_-)/S - (1-\lambda)\tilde{a}_{t-1,6}}{\lambda}\right]. \tag{30}$$

The yaw constraints clip this coordinate to the preimage and then to $[-1, 1]$. They act on a coordinate that the translational intervals do not involve, so Proposition 1 and its proof are unchanged.

Gear meshing. The held gear is gripped on its hub next to two free gears on the same row. Once it is below the tops of the neighbouring hubs, a finger line close to the direction of the row brings a finger into contact with a hub, and the release of the resulting moment produces force peaks that the translational intervals do not see. The keep-out keeps the commanded yaw out of this cone. Let ψ_h

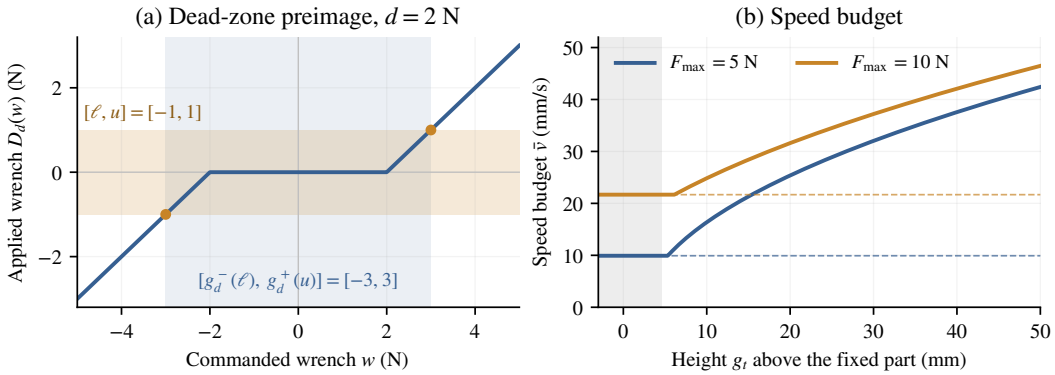

Figure 5: **Interval construction.** (a) Preimage of the applied-wrench interval $[-1, 1]$ N under a dead zone of $d = 2$ N (Lemma A.1); the commanded-wrench interval contains the whole zero-output band. (b) Speed budget (28) for peg insertion at 5 and 10 N with $\|v_t\|_2 = 0.01$ m/s; dashed lines mark the touchdown speeds, and the shaded region has zero effective gap. Both panels evaluate the formulas of this appendix.

be the current hand yaw, ψ_r the heading of the gear row and $\psi_f = \pi/2$ the direction of the finger line in the hand frame, and let $\vartheta = \mathrm{wrap}_{[-\pi/2,\pi/2)}(\psi_h + \psi_f - \psi_r)$ be the angle between the finger line and the row. The admissible target offsets are

$$[\psi_- - \psi_h,\ \psi_+ - \psi_h] = \begin{cases} [\alpha_{\min} - \vartheta,\ \pi - \alpha_{\min} - \vartheta], & \vartheta \geq 0, \\ [-\pi + \alpha_{\min} - \vartheta,\ -\alpha_{\min} - \vartheta], & \vartheta < 0. \end{cases} \tag{31}$$

Both arcs have width $\pi - 2\alpha_{\min}$ and keep the finger line at least $\alpha_{\min}$ from the row; inside the cone, the nearer edge demands motion out of it. Both gear CAP-WM configurations use $\alpha_{\min} = 42°$: the largest finger-line angle at which we measured a hub-contact torque, 37.0°, plus a closed-loop overshoot allowance of 4.8°, rounded up. The keep-out is active from the first decision of an episode and constrains the target yaw, which the rotational controller tracks.

Nut threading. An episode of nut threading succeeds only when the nut is threaded and the hand yaw is below zero, and the commandable yaw range leaves a fixed amount of screwing-in rotation per episode; rotation spent while the nut is still above the bolt reduces what remains for threading. The seat-first gate of the 5 N CAP-WM configuration therefore forbids screwing-in rotation while the held part is above the tip of the fixed part, $g_t > 0$, by imposing $\psi^{\mathrm{d}} \geq \psi_h$. By (29), this is

$$a_{t,6} \leq \frac{(O - \psi_h)/S - (1 - \lambda)\tilde{a}_{t-1,6}}{\lambda}, \tag{32}$$

and the gate releases the coordinate at $g_t \leq 0$. The other configurations use no yaw constraint.

B.5 Learning with the Projection in the Loop

Let $S_x(a)$ denote the complete execution map, including the fallback of Appendix A.4 and any yaw constraint. If the state s includes the controller memory, an agent that proposes a collects experience from the system

$$P_S(s' \mid s, a) = P\big(s' \mid s, S_{x(s)}(a)\big), \qquad R_S(s, a) = R\big(s, S_{x(s)}(a)\big). \tag{33}$$

Under partial observability the encoder represents the observation history; (33) specifies which action variable conditions the transition, independently of the quality of that representation.

World-model replay. CAP-WM stores the proposal a_t as the action of a transition and records the executed action beside it. The dynamics, reward and value heads and the policy prior are therefore trained on the proposal, the variable over which MPPI searches. The observation history receives the executed command instead, since that command drove the controller. The same convention holds in the seed phase and in the planning phase, and every environment action passes through the projection. The planner searches over proposal sequences in the learned model, and the projection is applied to the selected first action using the current controller state.

Policy likelihoods. For CAP-PPO, let a_t^{raw} be the Gaussian sample before the clip to the action range and the projection. The importance ratio is

$$r_t(\theta) = \frac{\pi_\theta(a_t^{\mathrm{raw}} \mid h_t)}{\pi_{\theta_{\mathrm{old}}}(a_t^{\mathrm{raw}} \mid h_t)}, \tag{34}$$

where h_t is the policy's observation history. For a fixed execution map, the trajectory density factors into policy densities of the raw samples and transition kernels of the form (33), and only the policy factors depend on θ. The logarithmic derivative therefore gives the usual score terms $\sum_t \nabla_\theta \log \pi_\theta(a_t^{\mathrm{raw}} \mid h_t)$, which justifies evaluating the ratio at the recorded raw sample while computing advantages from the rewards of the projected rollout. It also avoids assigning a Gaussian density to the executed action, whose distribution has mass on the interval boundaries.

C Experimental Details

This section describes the tasks, the hyperparameters of both backbones, the baselines, training and checkpoint selection, the metrics, and the compute used.

C.1 Tasks, Controller and Observations

We use the peg-insertion, gear-meshing and nut-threading tasks of FORGE (Noseworthy et al., 2025) in Isaac Lab, with FORGE's task-space impedance controller, rewards and success criteria. Episodes last 10, 20 and 30 s. The controller runs at 120 Hz and holds each command for eight physics steps, a 15 Hz decision rate. The seven-dimensional action contains translation, rotation and a success prediction; FORGE fixes roll and pitch, so translation and yaw are the physical control channels.

At every reset, the translational stiffness (nominally 565 N/m), the position-error clip (nominally 0.02 m) and the rotation-error clip (nominally 0.097 rad) are multiplied or divided, independently per axis, by a factor drawn uniformly from $[1, 1.41]$, $[1, 1.25]$ and $[1, 1.29]$, respectively. The smoothing factor is drawn from $\mathcal{U}[0.025, 0.1]$. Translational dead-zone widths are drawn from $\mathcal{U}[0, 5)$ N and rotational ones from $\mathcal{U}[0, 1)$ N·m, at reset and again every 2 s. The projection reads the realized values.

The 24-dimensional observation contains the fingertip position relative to the fixed part, the fingertip orientation quaternion, the linear and angular velocity, the wrist force, the force-penalty threshold of the FORGE reward (drawn per episode from $[5, 10]$ N) and the previous smoothed action. The observed force carries Gaussian noise with a standard deviation of 1 N per axis. The force budget and all metrics use the environment's low-pass filtered wrist force before this noise.

Wrist depth. The world-model agents receive a 96×96 depth image from a wrist-mounted camera with a 48 mm focal length. Metric depth is normalized as

$$I^{\mathrm{norm}} = \mathrm{clip}\left(\frac{I - 0.35\,\mathrm{m}}{0.25\,\mathrm{m}}; -1, 1\right). \tag{35}$$

The depth encoder has four convolutions with 32 channels, kernels $7, 5, 3, 3$, stride 2 and ReLU activations; its 512 output features are mapped linearly to the latent dimension and normalized with SimNorm. A random shift with padding 4 is applied in gradient updates only. The state and depth embeddings have equal size and are summed and passed through a fusion MLP, following Hansen et al. (2026). The vector input stacks the observation and the executed action over the last 3 decisions on peg insertion and the last 32 on gear meshing and nut threading (93 and 992 dimensions), together with the current depth frame.

C.2 Hyperparameters

Table 5 lists the hyperparameters of both backbones; the controller randomization is given in Appendix C.1. CAP-WM keeps the training objective, the model sizes and the planner of TD-MPC2 (Hansen et al., 2024) in the configuration of its 5M model and adds the depth encoder and fusion of Appendix C.1; it also keeps TD-MPC2's heuristic discount and the burst of updates at the end of the seed phase, during which uniformly random proposals are executed through the projection. CAP-WM also trains small force-prediction heads on the detached latent with a separate

optimizer; they do not affect the world model, the planner or the projection. CAP-PPO uses the recurrent actor of the FORGE PPO recipe, trained from random initialization with the projection active from the first iteration and the ratio (34).

Table 5: **Backbone hyperparameters.** Where a value differs by task, the entry reads peg / gear / nut. A vector step advances all 32 environments of a world-model run by one decision.

CAP-WM	
Latent / hidden / state-encoder width	512 / 512 / 256
State-encoder layers / SimNorm group size	2 / 8
Depth encoder	4 convolutions (32 channels, kernels 7, 5, 3, 3, stride 2)
Q-functions / Q-function dropout	5 / 0.01
Reward and value bins / range	101 / $[-10, 10]$ in symlog space
Discount	0.967 / 0.983 / 0.989 (TD-MPC2 heuristic)
Learning rate / encoder multiplier	3×10^{-4} / 0.3
Consistency / reward / value loss weights	20 / 0.1 / 0.1
Temporal loss discount ρ / target update rate	0.5 / 0.01
Gradient-norm clip / policy entropy coefficient	20 / 10^{-4}
Planning horizon / iterations / samples	3 / 6 / 512
Elites / policy-prior trajectories / temperature	64 / 24 / 0.5
Planning standard deviation (minimum, maximum)	$(0.05,\ 2)$
Batch size / gradient updates per vector step	256 / 8
Observation history (decisions)	3 / 32 / 32
Seed phase (vector steps)	1,000 / 1,500 / 2,250
Replay capacity (transitions)	10^6 / 10^6 / 5×10^5
State input	running normalization, clipped to ± 10
Parallel environments	32
CAP-PPO	
Actor	2-layer LSTM (1,024 units), then MLP $512, 128, 64$
Action distribution	Gaussian, state-dependent standard deviation
Discount / GAE parameter	0.995 / 0.95
PPO clip / epochs per iteration	0.2 / 4
Actor / critic learning rate	10^{-4} / 10^{-3}
Actor learning-rate schedule / KL threshold	adaptive / 0.008
Actor learning-rate bounds	$[10^{-6}, 10^{-2}]$
Entropy coefficient / gradient-norm clip	0 / 1
Iterations / environments / episodes per environment	200 / 128 / 1

C.3 Baselines

Unconstrained learners. PPO (FORGE) is the rl_games recipe distributed with Isaac Lab: a two-layer LSTM with $1{,}024$ units before an MLP with widths $512, 128, 64$ and ELU activations for the actor and the critic, discount 0.995, GAE parameter 0.95, learning rate 10^{-4} with the adaptive KL schedule (0.008), clip 0.2, four mini-epochs, minibatch 512 and 128 environments, trained for 233, 467 and 351 epochs with horizons of 128, 128 and 256 steps on peg, gear and nut. TD-MPC2 + depth is CAP-WM without the projection and the force-prediction heads, with identical world-model hyperparameters, observations and step budgets. Neither learner receives the force limit, so each is evaluated once and scored at both limits.

Constrained reinforcement learning. CPO, PID-Lagrangian and FOCOPS (Achiam et al., 2017; Stooke et al., 2020; Zhang et al., 2020) maximize the task reward subject to a budget d on the expected episode cost, with the per-step cost

$$c_t = (\|F_t\|_2 - F_{\max})_+ \, \Delta t, \tag{36}$$

whose undiscounted episode sum is the excess force of (37). Each member follows its reference implementation with its own networks and optimizer (Table 6), uses 128 environments for 200 iterations of one episode per environment, and keeps the FORGE reward. The budget of a task and force limit is 0.18 times the excess force of an unconstrained reference policy at that limit, which gives $d = 1$ and $0.064\,\mathrm{N{\cdot}s}$ on peg insertion, 8.296 and 0.536 on gear meshing, and 39.226 and

15.315 on nut threading, at 5 and 10 N. A cumulative budget does not bound the force at any single step: an episode whose only excess is one control step at $F_{\max} + b/\Delta t$ has cost b and satisfies any budget $d \geq b$. For each member we evaluate the checkpoint ranked best on training rollouts (by cost feasibility, then violation, success and return) and the final one. On nut threading the family is represented by FOCOPS.

Table 6: **Constrained-RL hyperparameters.** All members share the cost (36), 128 environments and 200 iterations. J_C: episode cost; λ, ν: Lagrange multipliers.

	CPO	**PID-Lagrangian**	**FOCOPS**
Policy network	LSTM $2 \times 1{,}024$, MLP $512, 128, 64$	MLP $512, 512$, LSTM 512	MLP $64, 64$
Value networks	MLP $512, 128, 64$	shared with the policy	MLP $64, 64$
Discount (reward / cost)	0.995 / 1	0.99 / 0.99	0.99 / 0.99
GAE (reward / cost)	0.95 / 1	0.97 / 0.97	0.95 / 0.95
Constraint on J_C	undiscounted	undiscounted	discounted (0.99)
Policy update	KL step 0.01, 10 CG iterations	clip 0.1, 8 epochs	KL bound 0.02, 10 epochs
Learning rate	line search (0.8, 15 steps)	10^{-4}	3×10^{-4}, linear decay
Multiplier	—	$K_P = 0$, $K_I = 1$, $K_D = 0$, $\lambda_0 = 0$	$\nu_0 = 0$, rate 0.01, $\nu \leq 2$
Other	damping 10^{-5}	cost scale 10	temperature 1.5

Safety layers. SC^3 (Wang et al., 2026) and FCSCC (Liang et al., 2023) filter the frozen PPO (FORGE) policy of each task, the controller of the first row of Table 2, at evaluation time and enforce $F_{\max}$ without a margin. SC^3 solves a quadratic program at every control step that projects the nominal action onto the half-space of a first-order control barrier function on the force, with a disturbance observer; it acts in contact. FCSCC projects the commanded wrench in closed form onto the half-space of a second-order exponential barrier on $F_{\max} - \|F_t\|_2$, using the task-space inertia read from the simulator, and acts in and out of contact. Both use a contact stiffness identified on the task. Because the filters are sensitive to this stiffness, we evaluate several settings of it on every task and limit (SC^3: the identified value scaled by 0.25, 1 and 4; FCSCC: the identified value, a stiffer prior and a rigid prior of 25.4 kN/m), with further settings on peg insertion, and report the best setting at each limit (Table 12).

C.4 Training Budgets and Checkpoint Selection

Table 7 summarizes the training protocol. The model-free rows use 128 parallel environments and the world-model rows 32, so at equal episode lengths the world-model rows receive one quarter of the model-free environment steps.

Table 7: **Training protocol** of the rows of Table 2. Environment steps are in millions and count every environment of a run. Selection: highest success (SR), highest safe success with ties broken by SR (SSR), or highest success among candidates without a violating episode (SR | 0 viol.).

	PPO (FORGE)	**Safe RL**	**Safety layer**	**TD-MPC2 + depth**	**CAP-PPO**	**CAP-WM**
Learner	PPO	constrained PG	none	TD-MPC2	PPO	TD-MPC2
Force constraint	none	cost budget	filter	none	projection	projection
Observation	proprio.	proprio.	proprio.	+ depth	proprio.	+ depth
Parallel envs.	128	128	—	32	128	32
Env. steps, peg	3.82	3.81	—	0.96	3.81	0.96
Env. steps, gear	7.65	7.65	—	1.92	7.65	1.92
Env. steps, nut	11.50	11.49	—	2.88	11.49	1.21 / 2.88
Selection	SR	SSR	SSR	SR	SR | 0 viol.	SR | 0 viol.

Checkpoint selection. Unconstrained learners are ranked by success and the constrained baselines by safe success at the column's force limit, with ties broken by success. For CAP we select the most successful checkpoint among those without a violating episode; in every cell this is also the

most successful checkpoint overall, so the zero-violation requirement never changes the choice. The candidates of a cell are the evaluated checkpoints of its run and, for the baseline families, every member and setting (Table 12). The world-model agents on gear meshing and nut threading are compared on separate validation episodes; the other cells are compared on the evaluation episodes. The zero-violation result of CAP$^{\blacklozenge}$ therefore does not stem from the selection: violating episodes are rare among all evaluated CAP$^{\blacklozenge}$ checkpoints and occur only in checkpoints that were not selected. The baselines are selected on the same episodes from more candidates, so selection does not favor CAP$^{\blacklozenge}$ in the comparison.

C.5 Evaluation Metrics

Each cell is evaluated on $N = 640$ episodes, five in each of 128 parallel environments. Let $s_e \in \{0, 1\}$ be the environment's success criterion at the final step of episode e, and let $f_{e,t} = \|F_{e,t}\|_2$ be the wrist force read at the end of control step t. Then

$$p_e = \max_t f_{e,t}, \qquad v_e = \mathbf{1}\{p_e > F_{\max}\},$$
$$\mathrm{SR} = \frac{1}{N}\sum_e s_e, \qquad \mathrm{SSR} = \frac{1}{N}\sum_e s_e(1 - v_e),$$
$$\mathrm{EF} = \frac{1}{N}\sum_e \sum_t (f_{e,t} - F_{\max})_+ \Delta t, \qquad \mathrm{PF} = \frac{1}{N}\sum_e p_e. \tag{37}$$

The number of violating episodes is $\sum_e v_e$; a force equal to the limit is admissible. For a fixed controller evaluated on independent episodes, a rate has binomial standard error at most $1/(2\sqrt{640}) = 1.98$ percentage points, and zero violating episodes bound the per-episode violation probability by $1 - 0.05^{1/640} \approx 0.47\%$ at one-sided 95% confidence. The twelve CAP$^{\blacklozenge}$ evaluations contain no violating episode in 7,680. If the cells have violation probabilities $q_1, \ldots, q_{12}$ with mean $\bar{q}$, the probability of this outcome is $\prod_c (1 - q_c)^{640} \le (1 - \bar{q})^{7680}$ by the inequality of arithmetic and geometric means, so $\bar{q} \le 1 - 0.05^{1/7680} \approx 0.04\%$ at one-sided 95% confidence.

Projection statistics. For each diagnostic flag, we compute the fraction of control steps flagged within an episode and average it over episodes. The intervention flag records a change of the translational action by the projection; the yaw constraints are counted separately. The saturation flag marks steps at which the final clip to the action range changed the output of the force clip, and the speed and target-clip infeasibility flags mark steps at which the executed action lies outside the corresponding interval on at least one axis.

C.6 Compute

Every training run used a single NVIDIA RTX 4090 or RTX A6000 GPU and took between 2 and 20 hours, including physics simulation, depth rendering and periodic evaluation; the reported cells used about 300 GPU-hours in total. The software is Python 3.11, PyTorch 2.7.0 with CUDA 12.8, Isaac Sim 5.1.0 and Isaac Lab 2.3.2.

D Additional Results

D.1 Learning Curves at 10 N

Figure 6 complements the 5 N learning curves of Figure 4 with the 10 N configurations. The unconstrained agents are trained once and scored at both limits, whereas CAP$^{\blacklozenge}$ and the Safe RL members are trained for each limit.

D.2 Distribution of Episode Peak Forces

For the episode peaks p_e of (37), define the empirical exceedance

$$\widehat{S}(F) = \frac{1}{N}\sum_{e=1}^{N} \mathbf{1}\{p_e > F\}. \tag{38}$$

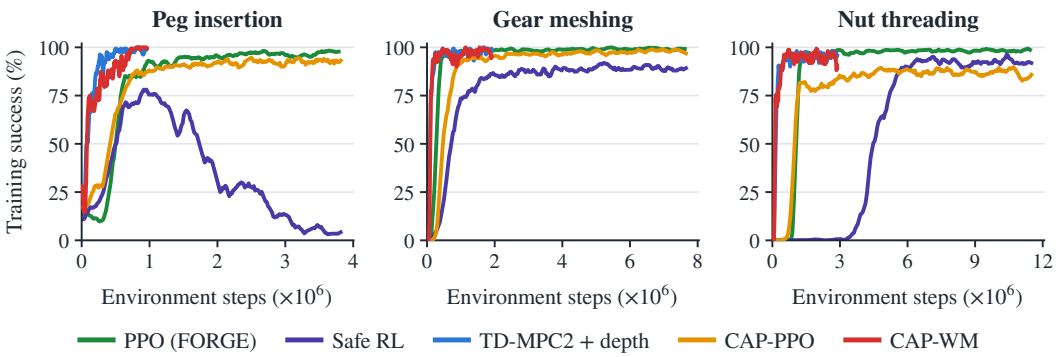

Figure 6: **Training success at the 10 N force limit.** Batch success, smoothed as in Figure 4; PPO (FORGE) and TD-MPC2 + depth repeat their unconstrained runs. The horizontal axis counts the interactions of all parallel environments. Table 2 gives the evaluation.

Its value at $F = F_{\max}$ is the fraction of violating episodes. Figure 7 shows $\widehat{S}$ for every row of Table 2. The CAP$^{\blacksquare}$ curves reach zero below the limit on all three tasks: the largest episode peak over the six CAP$^{\blacksquare}$ evaluations at each limit is 4.78 N at 5 N and 8.26 N at 10 N. The other curves show how the episodes of each baseline distribute around the limit, which the mean peak force of Table 2 summarizes in one number.

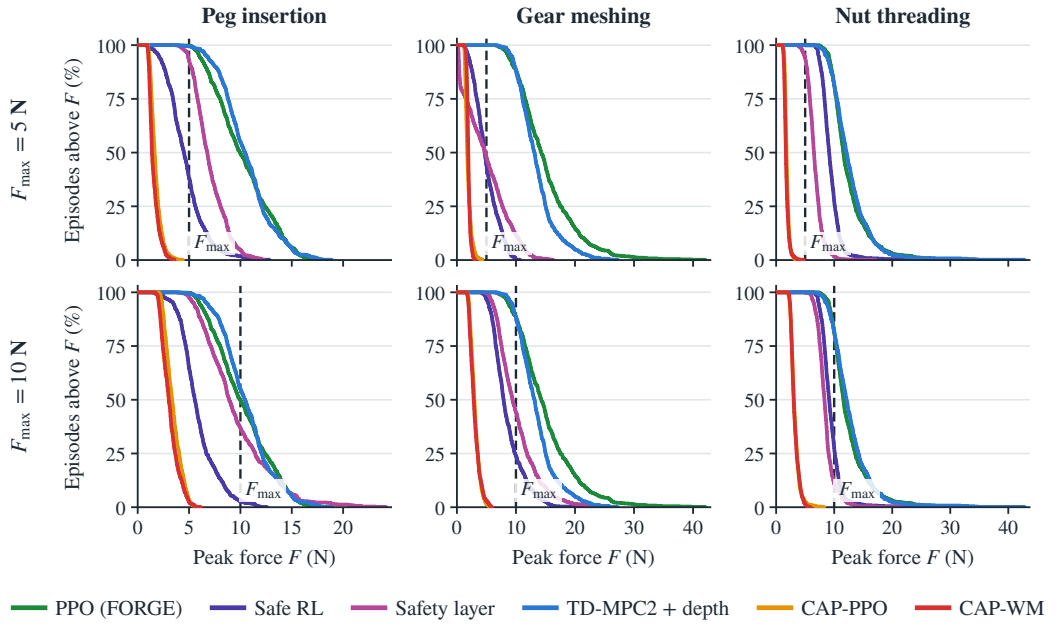

Figure 7: **Episode peak-force exceedance.** Fraction of the 640 evaluation episodes whose peak force exceeds the value on the horizontal axis (38). Columns are tasks and rows the two force limits; the height of a curve at the dashed line is the fraction of violating episodes.

D.3 Projection Activity

Table 8 reports the projection statistics of every CAP$^{\blacksquare}$ cell. The projection changes the translational action on 94.6 to 99.4% of control steps; a proposal outside a nonempty box is moved to its nearest admissible point, so frequent intervention does not indicate infeasibility. The final action-range clip decides on 1.6 to 6.7% of steps, and target-clip infeasibility does not occur in any evaluation, so by Corollary A.3 the bound on the modeled applied wrench holds on every other step. The yaw

constraints change the yaw command on 59.7% (gear, 5 N), 16.0% (gear, 10 N) and 14.9% (nut, 5 N) of steps.

Table 8: **Projection statistics** on the 640 evaluation episodes of Table 2, as within-episode step fractions averaged over episodes (Appendix C.5). Interv.: the translational action is changed. Yaw: the gear keep-out or the nut seat gate changes the yaw command. Sat.: the final action-range clip decides. Speed inf. and Clip inf.: the executed action lies outside the speed or target-clip interval on at least one axis. The flags are not mutually exclusive.

		Activity		Feasibility			
Task	**Method**	Interv. (%)	Yaw (%)	Sat. (%)	Speed inf. (%)	Clip inf. (%)	Viol. (/640)
Peg insertion	CAP-PPO🛡, 5 N	98.9	—	6.73	5.27	0.00	0
	CAP-PPO🛡, 10 N	97.6	—	2.92	2.85	0.00	0
	CAP-WM🛡, 5 N	97.8	—	6.17	5.26	0.00	0
	CAP-WM🛡, 10 N	94.6	—	2.39	2.36	0.00	0
Gear meshing	CAP-PPO🛡, 5 N	98.2	—	2.41	2.00	0.00	0
	CAP-PPO🛡, 10 N	98.4	—	1.59	1.59	0.00	0
	CAP-WM🛡, 5 N	96.5	59.7	2.98	2.59	0.00	0
	CAP-WM🛡, 10 N	98.3	16.0	2.12	2.12	0.00	0
Nut threading	CAP-PPO🛡, 5 N	99.0	—	3.65	3.61	0.00	0
	CAP-PPO🛡, 10 N	99.4	—	2.02	2.02	0.00	0
	CAP-WM🛡, 5 N	98.3	14.9	3.41	3.36	0.00	0
	CAP-WM🛡, 10 N	97.7	—	2.22	2.23	0.00	0

D.4 Execution Ablations with Fixed Checkpoints

Table 9 changes how trained controllers are executed while keeping their weights. Removing the projection from a trained CAP🛡 controller lowers safe success to at most 0.9% at both limits, and task success falls as well, on every task and for both backbones: the learned behavior relies on the projection that shaped it during training.

The second comparison places the unconstrained agents behind an evaluation-time force gate with the margin of the corresponding CAP🛡 configuration. While the wrist force read at the last decision is at or below $F_{\max} - \varepsilon$, the gate passes the proposed translation unchanged; above it, the gate replaces the translation by $\mathrm{clip}(-c/m; [-1, 1])$, the command for which the controller model predicts zero commanded wrench, and hands control back at the next decision at which the force is again below the threshold. At 5 N the gate reduces the excess force of PPO (FORGE) from 2.424 to 0.693 N·s on peg insertion, from 12.957 to 1.497 on gear meshing and from 36.907 to 1.565 on nut threading, yet at least 632 of 640 episodes still violate the limit. The gate reacts to contact that has already occurred, whereas CAP🛡 bounds the next command from the controller state and the approach speed before contact.

Table 10 evaluates the 5 N CAP-WM🛡 checkpoints of Table 2, trained with the planning horizon $H = 3$ of Table 5, with $H = 8$ and $H = 15$ at inference. Success depends on the horizon, and the training horizon is the best or within 1.4 points of it on every task. The projection bounds whatever the planner proposes: one of the 3,840 episodes exceeds the limit, with a peak of 5.13 N during a single control step, and that episode succeeds.

D.5 Wrist Depth and the Projection

Table 11 separates the two components of CAP-WM🛡 on peg insertion at 5 N: four world models on the same camera-equipped environment, with and without wrist depth and with and without the projection, all with an observation history of 3 and eight gradient updates per vector step. With the projection, depth raises success from 85.2% to 98.1%, and neither controller has a violating episode. Without the projection, depth raises success from 89.7% to 99.1%, while safe success stays at 0.6%.

Table 9: **Fixed-checkpoint execution ablations.** Each modified row reuses the checkpoint of the grey row above it, which repeats Table 2. None: the proposed action is executed without the projection or the yaw constraints. Force gate: the unconstrained agent behind the evaluation-time gate described in the text. No policy is retrained. **Bold**: no violating episode.

			Force limit $F_{\max} = 5\,\mathrm{N}$				Force limit $F_{\max} = 10\,\mathrm{N}$			
	Policy	**Execution**	**SR**↑ (%)	**SSR**↑ (%)	**EF**↓ (N·s)	**Viol.**↓ (/640)	**SR**↑ (%)	**SSR**↑ (%)	**EF**↓ (N·s)	**Viol.**↓ (/640)
Peg insertion	CAP-PPO🛡	projection	78.0	78.0	0.000	**0**	90.5	90.5	0.000	**0**
	CAP-PPO🛡	none	32.8	0.0	53.290	640	25.2	0.9	11.406	607
	CAP-WM🛡	projection	98.1	98.1	0.000	**0**	98.0	98.0	0.000	**0**
	CAP-WM🛡	none	92.0	0.0	59.947	639	78.8	0.6	22.118	634
	PPO (FORGE)	none	97.7	0.3	2.424	638	97.7	49.5	0.122	319
	PPO (FORGE)	force gate	94.1	1.2	0.693	632	97.0	48.3	0.123	322
	TD-MPC2 + depth	none	99.1	0.6	7.784	636	99.1	44.7	0.284	352
	TD-MPC2 + depth	force gate	94.1	0.9	1.069	634	98.9	49.1	0.108	323
Gear meshing	CAP-PPO🛡	projection	94.7	94.7	0.000	**0**	98.4	98.4	0.000	**0**
	CAP-PPO🛡	none	30.0	0.0	124.159	640	17.0	0.0	59.574	640
	CAP-WM🛡	projection	94.8	94.8	0.000	**0**	95.0	95.0	0.000	**0**
	CAP-WM🛡	none	34.8	0.0	141.262	640	50.8	0.0	48.326	639
	PPO (FORGE)	none	99.8	0.0	12.957	640	99.8	11.6	2.274	566
	PPO (FORGE)	force gate	88.9	0.0	1.497	640	97.5	17.8	0.338	525
	TD-MPC2 + depth	none	95.2	0.0	20.253	640	95.2	11.6	1.849	565
	TD-MPC2 + depth	force gate	88.3	0.0	1.533	640	94.5	21.4	0.248	501
Nut threading	CAP-PPO🛡	projection	61.4	61.4	0.000	**0**	87.2	87.2	0.000	**0**
	CAP-PPO🛡	none	38.0	0.0	108.685	640	27.2	0.0	74.597	639
	CAP-WM🛡	projection	89.2	89.2	0.000	**0**	91.9	91.9	0.000	**0**
	CAP-WM🛡	none	58.9	0.0	205.018	640	65.8	0.0	69.085	640
	PPO (FORGE)	none	97.5	0.0	36.907	640	97.5	18.6	5.465	520
	PPO (FORGE)	force gate	88.0	0.0	1.565	640	98.0	28.6	0.216	455
	TD-MPC2 + depth	none	96.2	0.0	43.143	640	96.2	17.7	3.907	525
	TD-MPC2 + depth	force gate	78.1	0.2	1.379	638	87.5	13.9	0.286	535

Table 10: **Planning horizon at inference.** The 5 N CAP-WM🛡 checkpoints of Table 2, trained with $H = 3$, evaluated with the horizon of each column on 640 episodes. The $H = 3$ column repeats Table 2. **Bold**: no violating episode.

	$H = 3$ (training)			$H = 8$			$H = 15$		
Task	**SR**↑ (%)	**SSR**↑ (%)	**Viol.**↓ (/640)	**SR**↑ (%)	**SSR**↑ (%)	**Viol.**↓ (/640)	**SR**↑ (%)	**SSR**↑ (%)	**Viol.**↓ (/640)
Peg insertion	98.1	98.1	**0**	94.7	94.7	**0**	79.7	79.7	**0**
Gear meshing	94.8	94.8	**0**	96.2	96.2	**0**	69.7	69.7	**0**
Nut threading	89.2	89.2	**0**	70.2	70.0	1	50.6	50.6	**0**

Depth improves task completion although the proprioceptive models were trained on 3.3 times as many environment steps, and the projection provides force compliance with or without depth.

Table 11: **Wrist depth and the projection** on peg insertion at $F_{\max} = 5\,\mathrm{N}$. Four TD-MPC2 world models on the same camera-equipped environment; the last two rows are the TD-MPC2 + depth and CAP-WM entries of Table 2. **Bold**: highest SSR and lowest EF.

Depth	**Projection**	Env. steps ($\times 10^6$)	**SR** ↑ (%)	**SSR** ↑ (%)	**EF** ↓ (N·s)	**PF** ↓ (N)	**Viol.** ↓ (/640)
—	—	3.20	89.7	0.6	6.530	10.53	636
—	✓	3.20	85.2	85.2	**0.000**	1.71	0
✓	—	0.96	99.1	0.6	7.784	10.60	636
✓	✓	0.96	98.1	**98.1**	**0.000**	1.56	0

D.6 Baseline Families

Table 12 lists every Safe RL member and safety layer behind the family rows of Table 2. The best member depends on the task and the limit: FOCOPS on peg insertion, PID-Lagrangian at 5 N and FOCOPS at 10 N on gear meshing, and FOCOPS on nut threading; among the safety layers, FCSCC at 5 N and SC^3 at 10 N on peg insertion and gear meshing, and FCSCC on nut threading. The family rows of Table 2 therefore report the strongest available member for each condition. For FCSCC, stiffening the prior lowers the number of violating episodes but also the success rate: on nut threading at 10 N, it goes from 424 violating episodes and 98.0% success at the identified stiffness to 71 and 77.5% with the stiff prior and 16 and 8.6% with the rigid one.

Table 12: **Baseline families.** For each Safe RL member, the better of its training-ranked and final checkpoints; for each safety layer, its best stiffness setting (Appendix C.3); both by SSR at each limit, ties broken by SR. Shaded: the entry reported in Table 2.

		$F_{\max} = 5\,\mathrm{N}$			$F_{\max} = 10\,\mathrm{N}$		
Task	**Member**	**SR** ↑ (%)	**SSR** ↑ (%)	**Viol.** ↓ (/640)	**SR** ↑ (%)	**SSR** ↑ (%)	**Viol.** ↓ (/640)
Peg insertion	CPO	52.7	23.4	358	26.9	23.9	88
	PID-Lagrangian	0.0	0.0	0	0.8	0.8	10
	FOCOPS	42.7	28.9	243	71.4	69.1	18
	SC^3	75.8	2.7	620	91.9	58.3	237
	FCSCC	90.2	6.1	594	97.0	53.6	289
Gear meshing	CPO	56.4	1.2	630	60.3	35.9	290
	PID-Lagrangian	31.6	16.2	281	55.9	9.4	398
	FOCOPS	57.3	2.7	620	75.8	58.3	153
	SC^3	67.2	1.2	631	87.3	52.7	283
	FCSCC	21.7	2.2	307	44.5	35.2	94
Nut threading	FOCOPS	90.2	0.0	640	90.2	64.8	177
	SC^3	86.9	0.2	639	94.2	35.3	409
	FCSCC	45.0	1.1	602	77.5	72.0	71