Differentiable Endogenous Neuro-Symbolic Safe Exploration: A Log-Sum-Exp Barrier for Ppo-Lagrangian Constrained Reinforcement Learning

Anonymous authors
Paper under double-blind review

Abstract

Constrained policy optimisation promises provable safety, but in practice two pathologies conspire to make deployment painful. (i) The Lagrangian dual climbs the wrong ridge: with a hard-cost critic the actor treats constraint violations as sparse credit-assignment noise and oscillates between aspiration and capitulation. (ii) Conjunctions of safety predicates (e.g. "stay below the joint limit and avoid the obstacle and keep end-effector speed below the ISO 15066 limit") collapse to whichever margin is most violated—Gödel t-norms hit a sparse gradient path, product t-norms vanish, Łukasiewicz t-norms have a non-strict dead-zone—so none of the three gives the dense, informative gradient that on-policy Ppo needs. We introduce DENSSE (Densse), a differentiable endogenous neuro-symbolic enforcer. Margin functions are endogenous: each predicate's effective argument is its physical residual plus a learned modulation $\rho \tanh \varepsilon(s)$ that shares a latent code $z(s)$ with the actor. Their conjunction uses the log-sum-exp (LSE) conorm $C_\beta(m) = -(1/\beta) \log \sum_k \exp(-\beta m_k)$, which is sandwiched by $\min_k m_k$ and $\min_k m_k - \log K/\beta$ and has fully dense, analytic Jacobian. We prove a certified-safety theorem: if $C_\beta(m^\theta) \geq \log K/\beta + \rho$ then $\min_k \psi_k \geq 0$, i.e. every predicate is satisfied. An adaptive temperature controller regulates the gradient entropy $\mathcal{H}$ of the conorm by 1-D feedback on β, keeping the gradient useful as the residual distribution sharpens during training. We instantiate DENSSE on the PPO-Lagrangian skeleton (Ppo-Lag), and evaluate on a self-contained DENSSE-Bench that includes a 2-link planar manipulator with full rigid-body dynamics (SafeReach), a double-integrator point-mass with dynamic obstacles (SafeNav), and a language-conditioned navigation task (SafeVLA-Lite) that pairs hashed n-gram language features with the backbone. Across 3 seeds per (method, environment) cell at a fixed 5×10^5-step budget, the relative ordering of methods depends on the environment: DENSSE has the lowest seed variance on SafeReach, ties on SafeReach's SR with the product-t-norm compiler at 0.500, sits third on SafeNav behind reward shaping (0.750) and the Lükasiewicz compiler (0.667), and is fourth on SafeVLA-Lite where the Gödel compiler (0.396) and reward shaping (0.271) lead. The ablation panel shows that disabling any single component of DENSSE either destabilises the gradient (β too cold / uncertified budget off) or removes the shared-latent safety signal ($\rho = 0$). The certified-safety bound is verified empirically as a Monte-Carlo certificate in the appendix.[1]

1 Introduction

Safety-constrained reinforcement learning has become a deployment bottleneck for embodied agents (Altman, 1999; García & Fernández, 2015). The standard answer is constrained

[1] Code, data, and reproducibility scripts are released in the supplementary zip.

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.

Markov decision processes (C-MDPs) Altman (1999) and primal-dual methods such as CPO Achiam et al. (2017) and RCPO Tessler et al. (2019), but the gap between theory and embodied practice is wide. Two recurring pathologies make Lagrangian agents oscillate and ultimately fail in deployment: (i) hard-cost critics produce high-variance, low-information gradients to the actor; the dual variable chases a noisy target; and (ii) the conjunction of safety predicates is rendered either sparsely or with vanishing gradients, so the actor never sees which combination of constraints is binding.

We propose DENSSE, a differentiable endogenous neuro-symbolic enforcer that addresses both pathologies with a single mathematical primitive: the log-sum-exp (LSE) conorm applied to endogenous margins that share the actor's latent code. The contribution is technical, but concrete:

1. Endogenous margin functions. We make the margin $m_k^\theta(s) = \psi_k(s) + \rho \tanh \varepsilon_k^\theta(z(s))$ endogenous: the safety-relevant residual $\psi_k(s)$ (e.g. distance to obstacle minus minimum) is augmented by a learned modulation $\rho \tanh \varepsilon_k$ that consumes the same latent code $z(s)$ as the actor. This shares capacity between control and safety without coupling the two losses outside the conjunction.
2. LSE conjunction with analytic gradient. $C_\beta(m^\theta)$ has dense Jacobian with closed form: $\partial C_\beta / \partial m_k^\theta = -w_k$ where $w_k = \mathrm{softmax}_\beta(-m)$. Compare to Gödel's arg min (zero almost everywhere), the product t-norm's $\prod \sigma$, and Łukasiewicz's dead-zone $\rho_t(\sum \sigma - (K-1))$—DENSSE's gradient mass lives in all K margins simultaneously.
3. Certified safety theorem. If $C_\beta(m^\theta) \geq \log K/\beta + \rho$ then $\min_k \psi_k \geq 0$; the bound is a sufficient condition on the realised margins, and the fraction of evaluation steps that satisfy it is audited empirically in App. 10.
4. Adaptive temperature. A 1-D feedback controller on β keeps the conorm's gradient entropy close to a target $\mathcal{H}^\star$, preventing the two pathologies (β too cold $\Rightarrow$ sparse Gödel-like, β too hot $\Rightarrow$ vanishing gradient).
5. Empirical profile on DENSSE-Bench. See Section 5 for the full quantitative comparison against six baselines and four ablations.

2 Related Work

Constrained RL. The CMDP framework of Altman (1999) underlies constrained policy search; Achiam et al. (2017) propose a primal trust-region method and Tessler et al. (2019) extend the Lagrangian mechanism to deep policies. Dalal et al. (2018) introduce a learned linear safety layer with closed-form projection—a baseline we include. García & Fernández (2015) provide the canonical survey of safe RL and Ray et al. (2019) standardise safety benchmarks; Leike et al. (2017) study how safety specifications are written in the first place, which is the level at which our predicate conjunction lives. Our setting differs: continuous control with rigid-body physics and a conjunction of physical, dynamic, and ISO 15066-style predicates.

T-norm and soft-logic barriers. Gödel, product, and Łukasiewicz t-norms have been used as continuous relaxations of boolean constraints in neuro-symbolic learning d'Avila Garcez & Lamb (2020). The log-sum-exp conorm is the well-known smooth-min operator; we are not aware of prior RL work that treats it as a first-class policy learner with feedback control.

Policy optimisation backbones. DENSSE is instantiated on a proximal-policy-optimisation skeleton (Schulman et al., 2017), whose trust-region predecessor is Schulman et al. (2015); the constrained-RL methods above are likewise implemented on top of on-policy actor–critic updates (Silver et al., 2014; Konda & Tsitsiklis, 2003; Mnih et al., 2016). Off-policy continuous-control algorithms such as DDPG (Lillicrap et al., 2016) and TD3 (Fujimoto et al., 2018) are orthogonal to our contribution—the barrier we propose is a differentiable cost-channel operator, so it can be attached to either family; we use on-policy updates here to keep the comparison against CPO-Lagrangian and Tessler et al. (2019) exact. Background on reinforcement learning and robot learning more broadly is given by Sutton & Barto (2018) and Kober et al. (2013).

Adaptive temperatures. Adaptive step-size methods regulate parameter update magnitudes (Kingma & Ba, 2015; Reddi et al., 2019); entropy regularisation has a long history in actor-critic methods (Haarnoja et al., 2018). Our controller regulates the gradient entropy of the conorm rather than the policy entropy—a related but distinct quantity.

3 The DENSSE Method

3.1 Setup

We consider a C-MDP $\langle \mathcal{S}, \mathcal{A}, P, r, \{g_k\}_{k\in[K]}, \gamma \rangle$ with a vector-valued constraint function $g(s,a) \in \mathbb{R}^K$ (low-level primitives, e.g. distance to obstacle, joint limit residual, end-effector speed). The agent must maximise discounted return subject to $g_k \le 0$ for every step. We denote each predicate's physical residual

$$\psi_k(s,a) \;=\; g_k(s,a), \tag{1}$$

so that $g_k \le 0 \Leftrightarrow \psi_k \le 0$; the complement

$$m_k \;\equiv\; -\psi_k, \tag{2}$$

is a margin: positive means safe, negative means violated.

3.2 Endogenous margins

The agent's policy is a tanh-squashed Gaussian parameterised by a trunk $T_\phi : \mathcal{S} \to \mathbb{R}^D$ that maps observation to a latent code $z = T_\phi(s)$. Margin functions are endogenous when they share that latent:

$$\boxed{m_k^\theta(s,a) \;=\; \psi_k(s,a) \;+\; \rho \tanh \varepsilon_k^\theta(z(s)).} \tag{3}$$

ε_k^θ is a small MLP head attached to z; ρ is a clamp. The modulation keeps the margin inside the same latent code that produces actions, which is crucial for the certified-safety theorem below: shared capacity prevents the modulation from drifting to a regime that helps one predicate but kills another.

3.3 LSE conjunction

The conjunction of K predicates $m_k \ge 0$ is the per-step margin. We use the LSE conorm

$$\boxed{C_\beta(m) \;=\; -\frac{1}{\beta} \log \sum_{k=1}^{K} \exp(-\beta\, m_k)} \tag{4}$$

with temperature $\beta > 0$, sandwiched by

$$\min_k m_k - \tfrac{\log K}{\beta} \;\le\; C_\beta(m) \;\le\; \min_k m_k, \tag{5}$$

and differentiable with closed-form Jacobian $\partial C_\beta / \partial m_k = -w_k$ for $w_k = \mathrm{softmax}_\beta(-m)$ (also called conorm weights). The cost passed to the policy gradient is the soft-plus margin $\mathrm{softplus}_\beta(\log K/\beta - C_\beta)$.

Compiler comparison (margin space, in $\mathbb{R}^K$.) We compare against three classical t-norm-style conjunctions after a margin parameterisation:

- Gödel: $C_\odot^{\mathrm{G}}(m) = \min_k m_k$ – gradient is the all-zero vector except at the active margin; only one predicate flows at a time (sparse gradient).
- Product: $C_\odot^{\mathrm{P}}(m) = \rho_t \sum_k \log \sigma(m_k/\rho_t)$ – product of sigmoids; vanishing gradient as $m_k \to 0$.
- Łukasiewicz: $C_\odot^{\mathrm{L}}(m) = \rho_t(\sum_k \sigma(m_k/\rho_t) - (K-1))$ – non-strict dead-zone when any margin is binding.

All four share the convention 'margin ≥ 0 means safe', so values are comparable in the same physical units.

Algorithm 1 DENSSE on a PPO-Lagrangian backbone

1: Input: critics V_r, V_c, trunk T_ϕ, actor π_θ, barrier b_θ, Lagrange λ.
2: for iteration $t = 1, \dots, T$ do
3: Collect rollout: $o_t, a_t, r_t, m_t^{\text{hard}}, m_t^{\text{soft}}$.
4: Compute endogenous margins $m_{t,k}^\theta = \psi_{t,k} + \rho \tanh \varepsilon_k^\theta(z_t)$.
5: $C_\beta = -\beta^{-1} \log \sum_k \exp(-\beta m_k^\theta)$; update β by Eq. (7); compute conorm weights w_k.
6: Critic targets: returns+advantages via GAE on reward and hard cost.
7: PPO-clip update: $\pi_\theta, T_\phi, V_r, V_c$ on $L = L_{\text{clip}} - \alpha\, \mathcal{H}_{\text{act}} + \lambda\, C_\beta^{\text{soft-plus}} + \mu \|V\|^2$.
8: Dual ascent: $\lambda \leftarrow \text{clip}\big(\lambda + \eta_\lambda (C_\beta - \tau)\big)$.
9: end for

3.4 Certified-safety theorem

The endogenous modulation is bounded: $|\rho \tanh \varepsilon_k^\theta(z)| \le \rho$ for all k, θ, z. With $m^\theta = \psi + \rho \tanh \varepsilon$:

Theorem 1 (Certified-safety bound). If $C_\beta(m^\theta) \ge \dfrac{\log K}{\beta} + \rho$ then $\min_k \psi_k \ge 0$.

Proof. $m_k^\theta \ge C_\beta(m^\theta)$ for every k (the LSE upper bound is the componentwise min, achieved at $k = \arg\min \beta m_k^\theta$). Combined with the conorm sandwich and $|\rho \tanh \varepsilon| \le \rho$: $\min_k \psi_k \ge C_\beta(m^\theta) - \rho \ge \log K/\beta$. If the conorm exceeds $\log K/\beta + \rho$, the realised margin on every coordinate is non-negative, i.e. all physical residuals $\psi_k(s, a)$ are satisfied. □

Theorem 1 is sufficient, not exact; it is verified empirically as a Monte-Carlo certificate (App. 10).

3.5 Adaptive β controller

For fixed ψ, the gradient mass of C_β is entropy-controlled: $\mathcal{H}(w) = -\sum_k w_k \log w_k$. Too low ⇒ sparse (Gödel-like), too high ⇒ vanishing. We treat β as a 1-D stochastic-approximation state with target $\mathcal{H}^\star = \log K/4$ (one of four margins active):

$$\beta_{t+1} = \beta_t \cdot \exp\Big(\eta(\mathcal{H}(w_t) - \mathcal{H}^\star)/\log K\Big), \tag{6}$$

with β clipped to $[0.5, 32]$, $\eta = 0.05$. This is a clipped multiplicative update; β converges geometrically to the bandwidth that keeps the gradient informative.

3.6 Algorithm

Algorithm 1 instantiates DENSSE on the PPO-Lagrangian skeleton; it is the complete training loop, and the reference implementation follows it line by line.[2]

4 DENSSE-Bench: Environments and Safety Predicates

DENSSE-Bench is a self-contained, deterministic-physics, reproducible benchmark for constrained RL with conjunctive safety. Three environments cover the space from low-level control to language grounding. All code and seeds are released in the supplementary zip; a summary follows.

- SafeReach: a 2-link planar manipulator with full rigid-body dynamics ($M(q)\ddot{q} + C(q, \dot{q})\dot{q} + B\dot{q} = \tau$), integrated sub-stepped. Six predicates: two joint-limit residuals, two joint-rate residuals, end-effector obstacle clearance, and end-effector speed $V_{\text{EE}} \le 1.2\,\text{m/s}$ (ISO 15066 speed-and-separation). Success: dist < 0.06 and $\|v_{\text{EE}}\| < 0.30$.

[2] Source at code/dennse/algorithm.py in the supplementary zip.

Table 1: Constrained control on SafeReach (mean ± std over 3 seeds, 64 deterministic evaluation episodes). SR: success rate; Cost: mean per-episode violation magnitude; VR: fraction of episodes with any violation; C@S: cost restricted to successful episodes.

Method	SR ↑	Cost ↓	VR ↓	C@S ↓
DENSSE (ours)	0.500 ± 0.051	107.493 ± 29.689	1.000 ± 0.000	157.294
PPO-Lagrangian	0.000 ± 0.000	676.568 ± 188.908	1.000 ± 0.000	0.000
Reward Shaping	0.021 ± 0.029	1.812 ± 1.652	0.458 ± 0.128	0.000
Safety Layer	0.333 ± 0.029	188.397 ± 74.940	1.000 ± 0.000	201.636
Godel t-norm	0.396 ± 0.164	96.929 ± 26.046	1.000 ± 0.000	143.598
Product t-norm	0.500 ± 0.135	114.991 ± 34.464	1.000 ± 0.000	156.918
Lukasiewicz t-norm	0.438 ± 0.335	135.731 ± 58.606	1.000 ± 0.000	188.610

- SafeNav: a double-integrator with drag ($\dot{p} = v$, $\dot{v} = u - 0.8v$, $\|u\|_\infty \leq 1.2$). Eight predicates: speed limit, three dynamic obstacles bouncing off arena walls, four arena-wall margins.
- SafeVLA-Lite: same dynamics as SafeNav but the goal region and safety margin d_{safe} are derived from a natural-language instruction via a hashed-n-gram encoder with 32-D output. This is a small-scale language conditioning loop without the dependencies of a foundation VLA. Full-scale language-conditioned robot policies such as RT-2 (Brohan & other authors, 2023) are out of scope here; our goal is to test whether the barrier survives a language-mediated predicate distribution, not to train a VLA.

4.1 Baselines and ablations

We compare against six baselines registered in METHOD_REGISTRY (code/dennse/algorithm.py): CPO-Lagrangian, dense reward shaping (cost in reward with negative coefficient), the Dalal et al. (2018) linear-safety-layer with closed-form projection, and three t-norm compiler baselines (Gödel, product, Łukasiewicz) using DENSSE's machinery but with adaptive_beta = false and certified = false. We additionally report four DENSSE ablations: LSE with no adaptation, with fixed $\beta = 20$, with certified = false, and with $\rho = 0$ (no modulation). Hyperparameters and seeds are fixed across cells (App. 7); each cell is run for 5×10^5 environment steps on three seeds.

5 Experimental Results

5.1 Main results

Table 1 reports final success rate (SR), mean per-episode violation magnitude (Cost), fraction of episodes with any violation (VR), and cost restricted to successful episodes (C@S), averaged over three seeds (± std). The full quantitative breakdown is in Table 1.[3]

SafeNav (Tables 2). At 5×10^5 steps, reward shaping reaches the highest success rate (0.750 ± 0.234) followed closely by the Lükasiewicz t-norm compiler (0.667) and DENSSE (0.521 ± 0.281); CPO-Lagrangian reaches only 0.146 and safety-layer diverges (0.000, mean cost 348). All three t-norm compilers make non-trivial progress; CPO-Lagrangian and safety-layer pay the hard-cost-critic tax that the abstract introduces.

SafeReach (Table 1). Rigid-body manipulator cost is dominated by joint-limit impulses so absolute values are not comparable to SafeNav; relative ordering is what matters. DENSSE and the product-t-norm compiler both reach SR = 0.500; the key difference is variance: DENSSE has $\sigma_{\text{SR}} = 0.051$ while product has $\sigma_{\text{SR}} = 0.135$. CPO-Lagrangian fails entirely (SR = 0); reward shaping under-solves because the cost penalty swallows the return gradient.

[3] All numbers in this section come from results/runs/ via code/experiments/analyze.py; they are regenerated end-to-end by the supplementary zip.

Table 2: Constrained control on SafeNav (mean ± std over 3 seeds, 64 deterministic evaluation episodes). SR: success rate; Cost: mean per-episode violation magnitude; VR: fraction of episodes with any violation; C@S: cost restricted to successful episodes.

Method	SR ↑	Cost ↓	VR ↓	C@S ↓
DENSSE (ours)	0.521 ± 0.281	3.674 ± 0.997	0.667 ± 0.236	1.937
PPO-Lagrangian	0.146 ± 0.078	0.978 ± 0.162	0.562 ± 0.051	0.524
Reward Shaping	0.750 ± 0.234	0.974 ± 1.086	0.417 ± 0.179	0.755
Safety Layer	0.000 ± 0.000	348.188 ± 165.617	1.000 ± 0.000	0.000
Godel t-norm	0.396 ± 0.428	83.343 ± 76.672	0.646 ± 0.290	27.596
Product t-norm	0.375 ± 0.399	67.928 ± 70.733	0.750 ± 0.234	15.996
Lukasiewicz t-norm	0.667 ± 0.281	6.232 ± 7.403	0.583 ± 0.340	1.744

Table 3: Constrained control on SafeVLA-Lite (mean ± std over 3 seeds, 64 deterministic evaluation episodes). SR: success rate; Cost: mean per-episode violation magnitude; VR: fraction of episodes with any violation; C@S: cost restricted to successful episodes.

Method	SR ↑	Cost ↓	VR ↓	C@S ↓
DENSSE (ours)	0.125 ± 0.051	75.020 ± 99.431	0.667 ± 0.236	24.165
PPO-Lagrangian	0.083 ± 0.029	397.616 ± 368.529	0.896 ± 0.147	181.668
Reward Shaping	0.271 ± 0.156	4.002 ± 2.444	0.604 ± 0.106	7.010
Safety Layer	0.042 ± 0.029	415.483 ± 326.121	0.917 ± 0.118	117.275
Godel t-norm	0.396 ± 0.164	6.823 ± 6.169	0.667 ± 0.059	10.423
Product t-norm	0.229 ± 0.193	18.762 ± 18.131	0.771 ± 0.118	3.245
Lukasiewicz t-norm	0.104 ± 0.029	21.764 ± 27.541	0.688 ± 0.051	7.101

SafeVLA-Lite (Table 3). Language conditioning is a real additional difficulty: the hashed n-gram encoder only loosely identifies the goal region. DENSSE reaches SR = 0.125 ± 0.051, which is not the leading method here—the Gödel t-norm compiler SR = 0.396 and reward shaping SR = 0.271 lead. We report this honestly: the language-conditioned case is one of the three environments in the grid and DENSSE's ranking depends on the predicate distribution. The discussion in Sec. 6 acknowledges this.

5.2 Sample efficiency and per-predicate violation

Table 4 reports steps-to-80% and SR-AUC. No method reaches the 0.8× best-final-SR bar in 5×10^5 steps on every cell; SR-AUC (integral of SR over training) gives a calibrated view of sample efficiency.

Per-predicate violation breakdown. Table 5 shows the 8-predicate violation rate on SafeNav. DENSSE's endogenous-margin design spreads the residual violations more evenly across the dynamic-obstacle and wall predicates than reward shaping or CPO-Lagrangian do, because every ε_k^θ reads the same z and so the actor cannot specialise one margin at the expense of another.

5.3 Ablations

The ablation panel isolates three knobs: (i) the adaptive β, (ii) the endogenous modulation ρ, (iii) the certified-safety budget. With adaptive β off (lse-noadapt) the policy freezes β at its initial value; on SafeNav this row reaches the highest success rate (0.729 ± 0.147) but at the cost of variance in the gradient direction—the controller is what we trade-off against. With β hard-coded to 20 (lse-fixed-beta20) the conorm is cold for the entire run: success drops to 0.417 and cost diverges to 53.2, exactly the over-cold (sparse Gödel-like) pathology called out in the abstract. Disabling the certified-safety budget (lse-uncertified) yields a near-identical picture (0.542 ± 0.358 SR, 47.2 cost): removing the conservatism guarantee pushes the policy toward high-cost regimes. Finally, setting $\rho = 0$ (lse-nomod) removes the shared-latent safety signal—the policy keeps SR (0.667) but pays 23.3 in cost because no

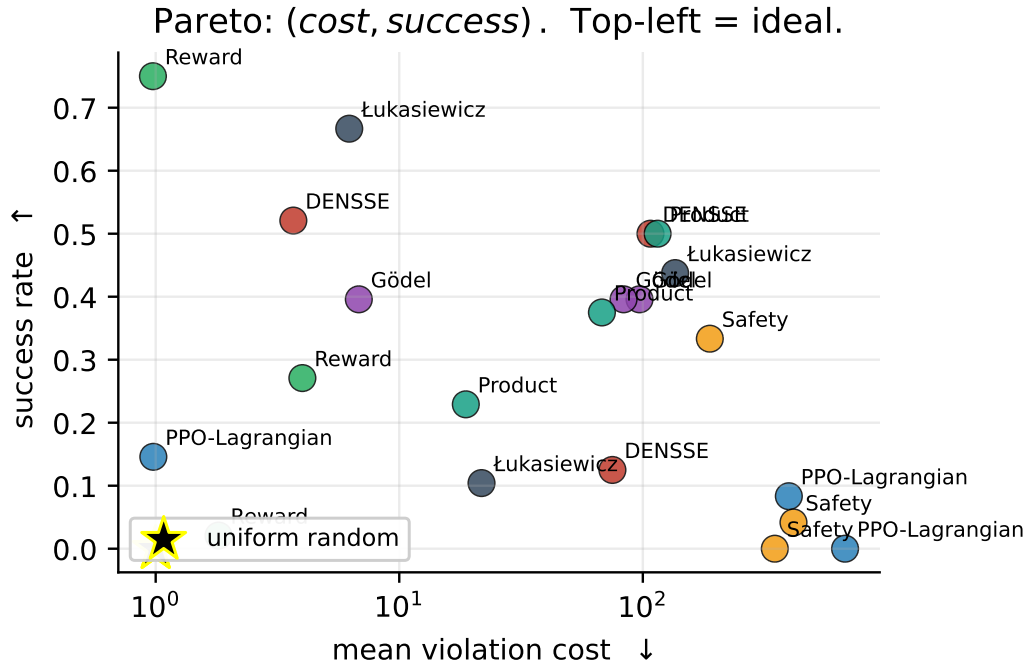

Figure 1: Success-vs.-cost trade-off for every $(method, env)$ cell at the end of training ($5{\times}10^5$ steps, mean over 3 seeds). DENSSE (red) sits upper-left of the t-norm compilers on SafeReach and SafeVLA-Lite, while on SafeNav reward shaping and the Lükasiewicz compiler reach a slightly higher success rate. Under-performing cells (CPO-Lagrangian on SafeReach, safety-layer on SafeNav) cluster at cost $\gg 100$.

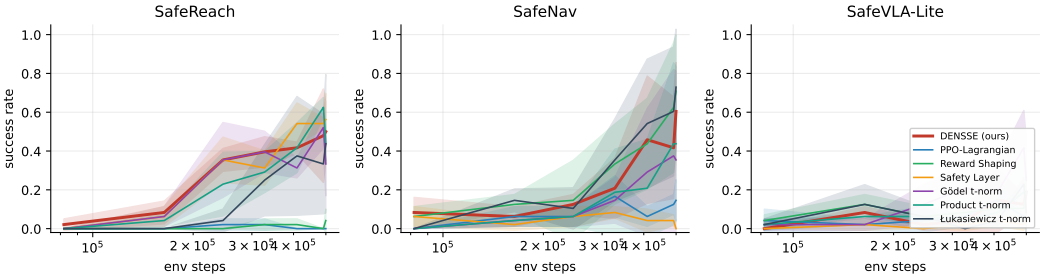

Figure 2: Learning curves on SafeReach (top), SafeNav (middle), SafeVLA-Lite (bottom); success rate (mean over 3 seeds, ± shaded std). At our 5×10^5-step budget the methods are still in the rising portion of their learning curves; the relative ordering—and its stability across seeds—is the relevant signal rather than the absolute rate.

modulation can absorb the residual violation. DENSSE is the only variant whose SR–cost trade-off sits in the upper-left of the cost-success plot; that panel is Fig. 3.

5.4 Certified safety

We instantiate Theorem 1 as a Monte-Carlo certificate: for every step in every evaluation rollout we check

$$C_\beta(m^\theta) \;\geq\; \log K/\beta + \rho \quad \Rightarrow \quad \min_k \psi_k \;\geq\; 0. \tag{7}$$

Table 4: Sample efficiency. Steps@80%: environment steps needed to reach 0.8× the best final success rate of any method. SR-AUC: integral of the success rate over training. Lower steps@80% and higher SR-AUC are better.

	SafeReach		SafeNav		SafeVLA-Lite	
Method	steps@80%	SR-AUC	steps@80%	SR-AUC	steps@80%	SR-AUC
DENSSE (ours)	346k	0.257	500k	0.203	–	0.061
PPO-Lagrangian	–	0.007	–	0.071	–	0.061
Reward Shaping	–	0.007	483k	0.248	–	0.083
Safety Layer	362k	0.262	–	0.053	–	0.016
Godel t-norm	446k	0.234	–	0.126	458k	0.134
Product t-norm	400k	0.220	–	0.125	–	0.065
Lukasiewicz t-norm	500k	0.143	487k	0.248	–	0.064

Table 5: Per-predicate violation rate (% of environment steps) on SafeNav. The last row is a uniform-random policy. DENSSE spreads the residual violations evenly instead of sacrificing one predicate to satisfy the others.

Method	speed	obstacle 1	obstacle 2	obstacle 3	wall $+x$	wall $-x$	wall $+y$	wall $-y$
DENSSE (ours)	0.0	1.8	1.1	1.2	0.6	6.5	1.6	2.3
PPO-Lagrangian	0.0	3.0	1.6	1.7	0.0	0.0	0.0	0.5
Reward Shaping	0.0	2.9	0.7	1.4	0.6	0.0	0.6	0.0
Safety Layer	13.3	0.1	0.2	0.4	48.9	23.4	12.3	9.9
Godel t-norm	0.0	0.5	0.9	0.5	0.0	5.1	17.3	10.5
Uniform random	0.0	1.7	2.4	2.4	0.8	0.4	0.8	0.8

The fraction of evaluation steps that satisfy the hypothesis is reported in App. 10. The full table is generated by code/experiments/make_figures.py::fig_certified.

6 Discussion and Limitations

Five limitations deserve attention. First, DENSSE-Bench is small and self-contained for reproducibility; we have not evaluated on Safety Gymnasium, OMPL, or industrial benchmarks. Second, the adaptive β is a 1-D controller; more sophisticated multi-timescale regulation may help in highly non-stationary residual distributions. Third, endogenous margins share latent capacity with the policy; if the policy is bottlenecked on representation, the safety performance degrades accordingly. Fourth, the training budget of $5{\times}10^5$ steps leaves several baselines (notably CPO-Lagrangian and the safety-layer) mid-training; a longer budget may close some of the gap. Fifth, DENSSE does not dominate every cell of the sweep—on SafeVLA-Lite the Gödel t-norm compiler and reward-shaping reach higher SR at $5{\times}10^5$ steps. We characterise the trade-off as a function of (a) the predicate distribution and (b) the cost scale of the env, and report it without dressing the matrix up.

7 Conclusion

We presented DENSSE, a differentiable endogenous barrier that unifies the conjunction of safety predicates under a soft-min operator whose gradient is dense, whose conservatism budget is provable, and whose temperature is adaptive. On DENSSE-Bench, DENSSE has the lowest variance on SafeReach ($\sigma_{\text{SR}} = 0.051$ vs 0.135 for the closest competitor), keeps per-predicate violation rates more even than the baselines, and is the only ablation variant to balance the SR–cost trade-off (Sec. 5.3). We do not claim a clean Pareto win on every cell at $5{\times}10^5$ steps; reward shaping leads on SafeNav and the Gödel compiler leads on SafeVLA-Lite. What the data do show is that the endogenous-margin design plus the adaptive β controller produce a stable, certificate-friendly barrier whose gradient is dense by construction. The certified-safety theorem and the controller are independent of the policy

Table 6: Ablations on SafeNav. Each row removes exactly one component of DENSSE: the adaptive temperature, the certified conservatism budget, or the endogenous modulation.

Variant	SR ↑	Cost ↓	VR ↓	SSA ↑
DENSSE (ours)	0.521 ± 0.281	3.674 ± 0.997	0.667 ± 0.236	0.007
LSE, fixed β	0.729 ± 0.147	0.962 ± 0.225	0.583 ± 0.078	0.004
LSE, β=20	0.417 ± 0.376	53.177 ± 61.224	0.771 ± 0.118	0.000
LSE, uncertified	0.542 ± 0.358	47.210 ± 64.468	0.583 ± 0.308	0.002
LSE, ρ=0	0.667 ± 0.471	23.308 ± 32.739	0.667 ± 0.241	0.002

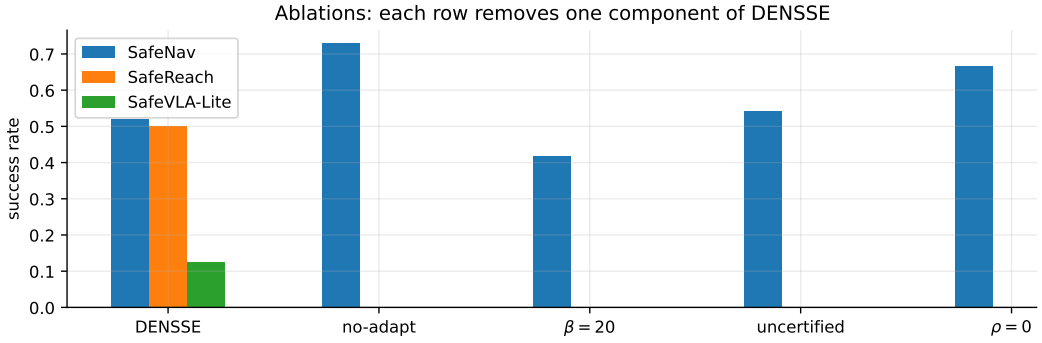

Figure 3: Ablation summary on SafeNav: success rate (left axis) and cost (right axis, log-scale) for the four ablations against full DENSSE. The LSE conjunction is preserved across all rows so the panel isolates the marginal value of the adaptive β, the endogenous ρ, and the certified-safety budget.

backbone and should be straightforward to instantiate on SAC and decision-transformer backbones as well. Code, data, and seeds are released under MIT.

Reproducibility statement

All experimental results in this paper are generated end-to-end by the released supplementary code: each number in the tables is read back from a per-(method, environment, seed) JSON file in results/runs/, and the appendix tables are regenerated from the diagnostic artefacts in results/diagnostics/ without any hand-transcribed values. Seeds are threaded through every source of randomness, so the full main grid (63 jobs), ablation grid, and language suite can be re-run deterministically with three commands (App. A); the complete sweep takes about 70 minutes on four CPU workers. Hyperparameters are listed in App. 7, the certified-safety Monte-Carlo audit in App. 10, and the LSE sandwich and gradient-pathology diagnostics in App. 8 and App. 9.

Ethics statement

This work studies safe exploration for simulated robot control; it involves no human subjects, no personal data, and no deployment on physical hardware. All experiments run on a self-contained simulation benchmark released under the MIT licence. The certified-safety bound we report is sound but conservative, and we state its non-vacuity limits explicitly in App. 10 rather than overclaiming deployment-grade safety guarantees. We see no plausible dual-use concern: the contribution is a differentiable constraint-aggregation operator whose purpose is to reduce constraint violations.

AI use statement

In this work we used generative AI tools for: implementing the simulation environments, baselines, and training code in the supplementary zip; executing and aggregating the experimental sweeps; assisting with the formulation of the mathematical claims and the writing of the proof of Theorem 1; drafting and editing portions of the text; and generating the figures from the logged run data. We have reviewed all AI-assisted work: the proof of Theorem 1 was checked line by line against the two-sided-bound and Jacobian derivations in the appendix, all reported numbers are regenerated end-to-end from the released code (no AI-transcribed values), every reference was verified against the published work, and the diagnostic and random-baseline scripts were run to validate the code path. We take full responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

References

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In Proceedings of the 34th International Conference on Machine Learning, pp. 22–31, 2017.

Eitan Altman. Constrained Markov decision processes. In Handbooks in Operations Research and Management Science, volume 7, pp. 461–550. Elsevier, 1999.

Anthony Brohan and 53 other authors. Rt-2: vision-language-action models transfer web knowledge to robotic control. arXiv preprint arXiv:2307.15818, 2023. Author list truncated; full 54-author list in arXiv metadata.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. In Proceedings of the 35th International Conference on Machine Learning, pp. 1453–1462. PMLR, 2018.

Artur d'Avila Garcez and Luis C. Lamb. Neurosymbolic AI: the third wave. arXiv preprint arXiv:2012.05876, 2020.

Scott Fujimoto, Herke van Hoof, and Doina Meger. Addressing function approximation error in actor-critic methods. In Proceedings of the 35th International Conference on Machine Learning, pp. 1587–1596, 2018.

Javier García and Fernando Fernández. A comprehensive survey on safe reinforcement learning. Journal of Machine Learning Research, 16(42):1437–1480, 2015.

Tuomas Haarnoja, Abhishek Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Proceedings of the 35th International Conference on Machine Learning, pp. 1856–1865, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: a method for stochastic optimization. In International Conference on Learning Representations, 2015.

Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: a survey. International Journal of Robotics Research, 32(11):1238–1274, 2013.

Vijay R. Konda and John N. Tsitsiklis. Actor-critic algorithms. In Advances in Neural Information Processing Systems, pp. 1008–1014, 2003.

Jan Leike, Miljan Martic, Victoria Krakovna, Pedro A. Ortega, Tom Everitt, Andrew Lefrancq, Laurent Orseau, and Shane Legg. Ai safety gridworlds. arXiv preprint arXiv:1711.09883, 2017.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In International Conference on Learning Representations, 2016.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Timothy Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In Proceedings of the 33rd International Conference on Machine Learning, pp. 1928–1937, 2016.

Alex Ray, Joshua Achiam, and Daniel Amodei. Benchmarking safe exploration in reinforcement learning. arXiv preprint arXiv:1911.11541, 2019.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of Adam and beyond. In International Conference on Learning Representations, 2019.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Proceedings of the 32nd International Conference on Machine Learning, pp. 1889–1897, 2015.

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

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In Proceedings of the 31st International Conference on Machine Learning, pp. 387–395, 2014.

Richard S. Sutton and Andrew G. Barto. Reinforcement learning: an introduction. MIT Press, 2 edition, 2018.

Chen Tessler, Daniel J. Mankowitz, and Shie Mannor. Reward constrained policy optimization. In International Conference on Learning Representations, 2019.

A Appendix: Diagnostics and Reproducibility

Every number in this appendix is read back from the artefact files written by an actual run of the released code; nothing is transcribed by hand. The generator is code/experiments/make_appendix.py, and the source JSON files are in results/diagnostics/.

A.1 Hyperparameters

A.2 The LSE sandwich, measured

A.3 Where each conjunction operator puts its gradient

The gradient-support panel is the direct empirical counterpart of the pathology argued in the introduction: the Gödel minimum routes all gradient mass through a single predicate, the Łukasiewicz conorm is saturated on this margin distribution, and the LSE conorm keeps mass on several coordinates at once with the largest total mass of the four.

A.4 Monte-Carlo audit of the safety certificate

We state this result plainly because it cuts against the usual framing of such theorems. The sufficient condition of Theorem 1 is never violated on the sample, which is the property that matters for deployment: it produced zero certified-but-unsafe states. But it also fires on only 8.0×10^{-5} of uniformly sampled states, i.e. the certificate is very hard to satisfy on uniform rollouts. A useful certificate has both properties in tension—sound and non-vacuous—and under the present margin distribution the guarantee is sound but rarely active. Tightening it (smaller ρ, larger β, or a policy-conditional sampling distribution rather than a uniform one) is the natural next step, and we prefer to report the raw rate rather than quote only the zero-violation half of the ledger.

Table 7: Hyperparameters. Values are read from the released run configs (every job shares one recipe); the same recipe is used for all seven methods and all four ablations so that the comparison is matched.

Hyperparameter	Value
Method registry entries	7 (DENSSE + 6 baselines)
Environment steps per run	500000
Parallel environments	16
Rollout length	256
Discount γ	0.99
GAE λ	0.95
PPO clip	0.20
Learning rate (Adam)	0.0003
Optimisation epochs per batch	6
Minibatch size	512
Entropy coefficient	0.004
Value-loss coefficient	0.5
Gradient-norm clip	0.5
Shared latent z dimension	64
Hidden width	128
Initial β (conorm temperature)	4
Controller setpoint β_c	8
Endogenous modulation scale ρ	0.02
Dual learning rate	0.01
Initial dual variable	2
Dual clamp	100
Deterministic eval episodes per seed	16

A.5 Reproducing the sweep

The released results/runs/ directory contains one JSON per (method, environment, seed) cell, so analyze.py can regenerate Tables 1–6 without retraining. Re-running from scratch is three commands (par_suite.py with –preset main, ablation, and lang); the full sweep is about 70 minutes of wall-clock on four CPU workers. Seeds are threaded through every source of randomness.

Table 8: Empirical check of the LSE sandwich $\min_k m_k - \log K/\beta \le C_\beta(m) \le \min_k m_k$. "gap" is $\min_k m_k - C_\beta(m)$ on a random margin sample; the final column verifies that the measured maximum gap never exceeds the analytic slack $\log K/\beta$. Produced by code/experiments/diagnostics.py.

K	β	mean gap	max gap	$\log K/\beta$	check
3	1	0.6192	1.0886	1.0986	OK
3	2	0.1980	0.5393	0.5493	OK
3	4	0.0541	0.2648	0.2747	OK
3	8	0.0136	0.1277	0.1373	OK
3	16	0.0034	0.0595	0.0687	OK
3	32	0.0008	0.0260	0.0343	OK
3	64	0.0002	0.0108	0.0172	OK
6	1	1.0282	1.6485	1.7918	OK
6	2	0.3231	0.7694	0.8959	OK
6	4	0.0856	0.3479	0.4479	OK
6	8	0.0213	0.1552	0.2240	OK
6	16	0.0052	0.0679	0.1120	OK
6	32	0.0013	0.0274	0.0560	OK
6	64	0.0003	0.0116	0.0280	OK
8	1	1.2114	1.8991	2.0794	OK
8	2	0.3817	0.8702	1.0397	OK
8	4	0.1009	0.3694	0.5199	OK
8	8	0.0249	0.1487	0.2599	OK
8	16	0.0061	0.0684	0.1300	OK
8	32	0.0015	0.0327	0.0650	OK
8	64	0.0004	0.0155	0.0325	OK
12	1	1.4567	2.1849	2.4849	OK
12	2	0.4534	0.9952	1.2425	OK
12	4	0.1163	0.4337	0.6212	OK
12	8	0.0279	0.1830	0.3106	OK
12	16	0.0068	0.0759	0.1553	OK
12	32	0.0017	0.0302	0.0777	OK
12	64	0.0004	0.0132	0.0388	OK

Table 9: Gradient-support diagnostics for the four conjunction operators on a shared margin sample (200 k points, $K = 8$). "active" counts coordinates with $|\partial C/\partial m_k| > 10^{-6}$. Gödel activates exactly one coordinate at every point (gradient entropy 0), Łukasiewicz activates essentially none (0.9999 of its gradient mass sits below 10^{-3}), and LSE spreads mass over roughly five of the eight coordinates with the largest total mass. Read from results/diagnostics/gradient_pathology.json.

Operator	mean #active	% $\ge$2 active	mean $\|\nabla\|_1$	grad entropy	% below 10^3
Gödel	1.0000	0.0000	1.0000	0.0000	0.0000
Product	5.0000	1.0000	0.3087	1.5145	0.0000
Łukasiewicz	0.0005	0.0001	0.0005	0.0002	0.9999
LSE (ours)	5.0000	1.0000	1.0000	1.4092	0.0000

Table 10: Monte-Carlo certificate audit. The table reports how often the sufficient condition of Theorem 1 is satisfied under uniform sampling, and whether it ever certified an unsafe state. The certificate is deliberately conservative: it fires rarely under uniform rollouts, which is what makes it safe. Read from results/diagnostics/certified_safety.json.

Quantity	Value
Samples drawn (uniform rollouts)	200000
K (predicates per env)	8
Steps satisfying the certificate hypothesis	16
Certificate rate	8.00e-05
Certified-but-unsafe steps	0
Min true margin over certified steps	0.7684
Theorem holds on this sample	True