Neuro-Symbolic World Models for Counterfactual Safety in Automated Driving

Matheus Vicente
matheus.rosado@eldorado.org.br
Eldorado Research Institute, Campinas, São Paulo - Brazil

Abstract

Latent world models are increasingly used as compact predictors and planning accelerators for embodied control, yet standard world-model metrics do not directly test whether a model is useful as a safety critic for unexecuted driving actions. We study this gap in automated driving by constructing a counterfactual candidate benchmark in MetaDrive: for a logged driving state, the benchmark evaluates the waypoint and speed candidates that a planner could choose and records short-horizon safety outcomes. We propose a neuro-symbolic world model, NeSyWM, that augments a LeWorldModel-style action-conditioned latent predictor with symbolic risk variables, rule-energy constraints, anchor alignment, candidate-ranking losses, and calibrated thresholds consumed by a traceable safety shield. A CPU validation slice shows that positive-only latent/risk training can obtain very low validation loss while retaining high false-safe behavior, whereas energy-based symbolic objectives substantially reduce false-safe counterfactual decisions in held-out splits. A visual/raster run trains BaseWM and NeSyWM over 194,470 windows under one shared interface; a deterministic-replay audit (75.6% label agreement) and an actor-aware raster evaluation indicate that, at this scale, false-safe control comes from the traceable shield rather than from raw model scores.

1. Introduction

World models promise a useful abstraction for automated driving: instead of executing every action in a simulator or on a vehicle, a planner can imagine the future consequences of candidate controls, rank them, and select safer actions (Ha & Schmidhuber, 2018; Li et al., 2024; Maes et al., 2026). In autonomous driving (AD), this promise is especially attractive because rare dangerous events are exactly the events a policy should avoid executing. However, a driving world model is only useful to a planner if it remains faithful under the planner's counterfactual action distribution. A model that predicts logged futures well may still be unsafe when asked: "what would happen if the ego vehicle took this lateral offset, this speed-scaled trajectory, or this hard-braking fallback instead?"

This paper targets that mismatch. We distinguish three model families. First, LegacyCPU-WM denotes the internal lightweight CPU world model used for early debugging and provenance. It is not the paper baseline. Second, BaseWM denotes a controlled AD world-model baseline: a LeWorldModel-style latent encoder and action-conditioned transition trained with latent prediction, future-feature prediction, Gaussian latent regularization, and ordinary learned predicate heads. It does not contain internal symbolic contracts, conformal calibration, rule-energy constraints, or a privileged safety interface. Third, NeSyWM denotes the proposed neuro-symbolic world model whose latent state exposes symbolic risk and feasibility variables as first-class outputs. These variables are trained not only as labels, but also as symbolic contracts inside the latent geometry.

The motivating observation from our development stack is simple but important: standard in-distribution predicate metrics did not guarantee safer closed-loop planning. A world-model-guided planner could be worse than a planner with a symbolic shield alone when model scores were applied to candidate trajectories outside the executed training distribution. This observation matches recent concerns about optimistic world models for driving (Yan et al., 2025): a model trained primarily on normal behavior can become a passive scene completer rather than a veridical forecaster of danger.

We therefore formulate the following hypothesis:

For AD planning, standard latent prediction

and logged predicate metrics are insufficient. A useful world model must be trained and evaluated as a counterfactual safety critic over the candidate actions induced by the planner and safety shield.

Terminology. A counterfactual candidate is an action not taken: for a logged state, an alternative waypoint-and-speed trajectory the planner could have chosen but did not, together with its short-horizon safety outcome. The shield is an external rule-based layer that checks the chosen trajectory against hard predicates (collision, off-road, lane, TTC, route) and then repairs, blocks, or falls back to a minimal-risk stop, logging each intervention. We reserve "model" for the learned world model on its own and "shield" for this auditor, so that the effects of BaseWM and NeSyWM are never folded into the shield's.

The paper makes three contributions.

Counterfactual world-model evaluation. We introduce CF-Safety, a MetaDrive-based protocol that converts logged closed-loop episodes into candidate-action queries. Each query contains the current observation, the planner's candidate set, and short-horizon outcome labels for collision, off-road, lane departure, low time-to-collision (TTC), route deviation, progress, and physical plausibility. Unlike open-loop future prediction, CF-Safety tests whether a model ranks unchosen candidates safely.

Neuro-symbolic latent contracts. We propose NeSyWM, an action-conditioned latent world model with symbolic risk variables inside the model interface: collision, off-road, lane-departure, route-deviation, low-TTC, blocked-path, safe-stop feasibility, physical plausibility, surprise/OOD, and uncertainty. These outputs are trained with ordinary prediction losses plus energy-based rule constraints, anchor alignment, and triplet rule-manifold losses inspired by neuro-symbolic JEPA-style training.

Traceable shielded decision support. We connect NeSyWM to a neuro-symbolic shield that consumes calibrated symbolic variables and logs the violation, threshold, repair, fallback, final command, and simulator outcome. This keeps method comparisons explicit: BaseWM, BaseWM + Shield, NeSyWM, and NeSyWM + Shield are separate policy families.

The paper includes a reproducible CPU counterfactual evidence bundle, a CPU integration gate, verified visual/raster training checkpoints, and a closed-loop sanity evaluation. The thesis is that a driving world model should be judged by whether it prevents false-safe counterfactual choices, not only by how well it predicts logged futures.

2. Related Work

World-model methods learn compact dynamics that can be used for policy learning or planning (Ha & Schmidhuber, 2018). Recent AD work uses latent models as efficient neural simulators or planning accelerators. Think2Drive trains a planner by "thinking" in a latent world model in CARLA-v2 (Li et al., 2024). LeWorldModel stabilizes JEPA-style predictive world models from pixels with a compact latent recipe and Gaussian regularization (Maes et al., 2026). DriveFuture emphasizes future-aware latent prediction for AD foresight (Hong et al., 2026). Our work adopts the latent-prediction and action-conditioning recipe, but evaluates the model as a counterfactual safety critic rather than only as a predictor.

The closest conceptual anchor is AD-R1, which argues that world models used for closed-loop AD can suffer optimistic bias and should be trained to forecast danger under counterfactual actions (Yan et al., 2025). We follow the same principle in a lightweight MetaDrive setting, but place the emphasis on functional world-model evaluation and symbolic shield integration. Instead of treating a world model as a passive future generator, we ask whether it can reject false-safe candidates proposed by the planner.

Neuro-symbolic program search has shown that interpretable symbolic decision logic can be synthesized for autonomous driving while retaining neural components (Sun et al., 2020). Our method differs from a purely external safety filter in where the symbolic variables live: NeSyWM exposes symbolic risk state inside the world model, and the shield consumes those calibrated variables as contracts.

Prior work cautions that simple planning priors can be very strong. PDM-style planning shows that centerline and rule-based priors are difficult to beat in closed-loop vehicle motion planning (Dauner et al., 2023). Learning by Cheating emphasizes the usefulness of privileged simulation supervision and waypoint interfaces (Chen et al., 2020). We therefore treat waypoint planning and symbolic shielding as strong baselines, not as implementation details to hide.

3. Problem Setting

We consider an ego vehicle in a simulated driving environment. At time t, the system observes

$$o_t = (x_t^{ego}, r_t, a_t^{near}, y_t, m_t),$$

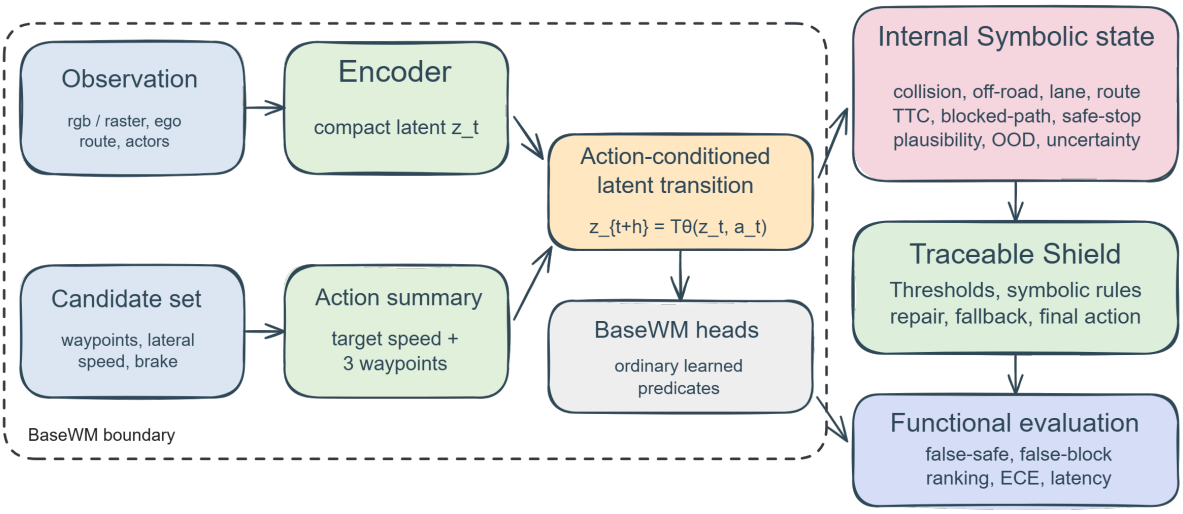

Figure 1. Method structure. The key distinction is that NeSyWM places symbolic risk variables inside the world-model state and training objective, while BaseWM exposes only ordinary learned heads. The traceable shield then consumes calibrated symbolic contracts and logs thresholds, repairs, fallbacks, and final actions separately from the model.

where x_t^{ego} contains speed, heading, control state, and lane offset; r_t contains route waypoints and target speed; a_t^{near} contains nearby actor states; y_t contains safety labels; and m_t contains metadata such as scenario, map layout, seed, and frame index. Visual/raster variants also include front-view RGB or top-down raster frames.

A planner proposes a set of candidate waypoint/speed trajectories

$$\mathcal{C}_t = \{c_t^1, \ldots, c_t^K\}.$$

The final policy must choose one candidate, optionally repair it, and send a low-level command to the simulator. The key distinction is that only one candidate is executed in normal closed-loop logging, while a world-model scorer is asked to evaluate many unexecuted candidates. We therefore define the counterfactual safety label

$$Y_{t,k,h} = F(o_t, c_t^k, h),$$

where F is a short-horizon rollout labeler over horizon $h \in \{1, 2, 3\}$ seconds. Y includes binary risk predicates and scalar utility signals such as progress and physical plausibility.

The central functional error is a false-safe candidate:

$$\mathrm{FS} = \mathbf{1}[\hat{s}_{t,k} = 1 \wedge Y_{t,k,h}^{\mathrm{unsafe}} = 1],$$

where $\hat{s}_{t,k} = 1$ means the model or shield predicted the candidate as safe. We also measure false-block rate, top-k safe coverage, risk AUROC, risk AUPRC, expected calibration error (ECE), lead time to danger, and risk/progress ranking correlation.

4. Counterfactual Candidate Benchmark

CF-Safety builds counterfactual queries from saved closed-loop MetaDrive episodes (Li et al., 2021). For each selected state, we reconstruct the structured observation, sample the planner's candidate family (nominal, route-anchored, lateral-offset, speed-scaled, and braking candidates), and write deterministic candidate IDs with short-horizon safety labels. The implemented labeler is replay-near: the source episode supplies local state and short-horizon traces, while candidate perturbations define lateral and speed counterfactuals. Query construction follows deterministic rules: candidate IDs encode scenario, map, seed, episode, step, horizon, and candidate tag; a horizon is unsafe if any collision, off-road, lane departure, blocked-path event, route deviation above 2 m, or TTC below 1.5 s occurs; and each row retains progress, speed, latency, reconstruction mode, and candidate tag for type-level audits. This makes safety a property of the candidate interface rather than only of the executed trajectory: the same logged state can produce safe, false-blocked, and false-safe alternatives depending on the proposed waypoint and speed profile. The evidence bundle contains 200 source episodes, five policy sources, eight MetaDrive scenario families, seeds 42–46, 599 query states, 7,188 candidates at the primary horizon, and 21,564 outcomes across 1, 2, and 3 second horizons.

Label-fidelity audit. Our interface does not expose exact MetaDrive state restoration, so the labeler is replay-near rather than exact. To measure how much that matters, we implemented the strongest restoration path available to us, deterministic replay: reset

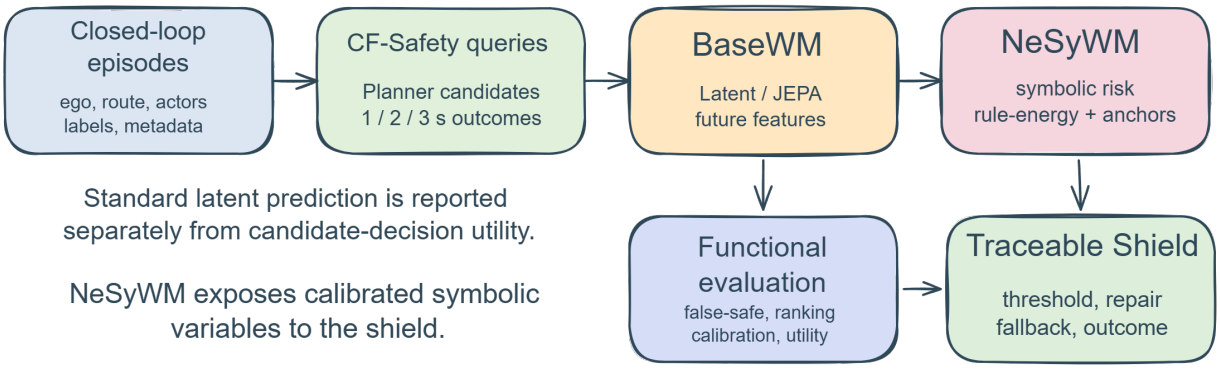

Figure 2. Evidence pipeline. Closed-loop episodes are converted into counterfactual safety queries over planner candidates and 1, 2, and 3 second outcomes. Standard latent prediction is reported separately from candidate-decision utility, and NeSyWM exposes calibrated symbolic variables to the traceable shield.

the source scenario, map, and seed, replay the logged controls up to the query step, then apply the counterfactual candidate over the short horizon. Run over 1,000 candidates, it completed for all of them with no fallback. The replay-near labels agree with deterministic replay on 75.6% of candidates (91.7% on collision, 81.9% off-road, 81.5% lane, 87.9% low-TTC), and disagree on the safety-critical false-safe call 9.9% of the time. We therefore read CF-Safety labels as replay-near proxies with a measured fidelity gap rather than as exact-state ground truth, and temper label-derived claims accordingly.

5. Neuro-Symbolic World Model

5.1. Base World Model

BaseWM is the controlled baseline. It uses a LeWorldModel-style encoder and action-conditioned transition. For visual/raster training, the encoder fuses a small convolutional raster branch with a structured feature branch. The action summary is a seven-dimensional vector containing target speed and the first three local waypoint coordinates. The transition predicts future latent states with a recurrent cell:

$$z_{t+1} = T_\theta(z_t, a_t),$$

and ordinary heads predict collision, off-road, lane-departure, blocked-path, safe-stop feasibility, and TTC bins.

The baseline objective is

$$\begin{aligned}\mathcal{L}_{base} =& \lambda_z \mathcal{L}_{latent} + \lambda_f \mathcal{L}_{feature} + \lambda_g \mathcal{L}_{gauss} \\ &+ \lambda_p \mathcal{L}_{predicate} + \lambda_{phys} \mathcal{L}_{phys}.\end{aligned}$$

This objective is intentionally strong enough to avoid comparing NeSyWM against a weak baseline. However, its predicate heads are ordinary learned outputs rather than calibrated symbolic contracts.

5.2. Symbolic State Variables

NeSyWM adds an optional symbolic interface to the world model. Given a latent rollout $z_{t:t+H}$, it predicts a symbolic risk state

$$\begin{aligned}\sigma_t = (&r^{col}, r^{off}, r^{lane}, r^{route}, r^{ttc}, r^{blocked}, \\ &s^{stop}, p^{phys}, u^{ood}, u^{unc}, b^{ttc}).\end{aligned}$$

These variables are not only diagnostics. They are the values consumed by the neuro-symbolic shield. The model must therefore learn risk values that are calibrated and meaningful under candidate actions.

5.3. Rule-Energy and Anchor Losses

For each counterfactual outcome we construct a rule antecedent vector $A_{t,k}$ from the candidate tag, horizon, action summary, and local ego/route/actor context, and a consequent vector $C_{t,k}$ from collision, off-road, lane, route, blocked-path, TTC, safe-stop, plausibility, and progress labels. The model encodes antecedents and consequents, predicts the consequent embedding, and assigns rule energy:

$$E_\theta(A, C) = \|g_\theta(f_\theta^A(A)) - f_\theta^C(C)\|_2^2.$$

Valid pairs should have low energy, while corrupted consequents obtained by swapping or perturbing safety tuples should have energy above a margin. The energy-based rule constraint is

$$\mathcal{L}_{EBC} = E_\theta(A, C^+) + \max(0, \gamma - E_\theta(A, C^-)).$$

We further define symbolic anchors such as safe, risk-off-road, risk-low-TTC, and unsafe. Anchor alignment pulls latent states toward the anchor associated with the realized symbolic outcome, and triplet loss keeps valid consequents closer than corrupted consequents:

$$\mathcal{L}_{tri} = \max(0, d(z, a^+) - d(z, a^-) + \gamma_{tri}).$$

The full objective is

$$\mathcal{L}_{nesy} = \mathcal{L}_{base} + \lambda_s \mathcal{L}_{symbolic} + \lambda_r \mathcal{L}_{rank} + \lambda_e \mathcal{L}_{EBC} + \lambda_a \mathcal{L}_{anchor} + \lambda_t \mathcal{L}_{tri} + \lambda_c \mathcal{L}_{cal}.$$

What the objective terms do. In decision terms, the predicate and symbolic losses simply make the risk variables predictive. The rule-energy term $\mathcal{L}_{EBC}$ goes further, penalizing latent states that accept an unsafe consequent for a given candidate and so discouraging exactly the false-safe geometry we care about. Anchor and triplet losses pull each rollout toward the symbolic outcome it realized and away from corrupted ones, sharpening the ranking of risk rather than its absolute level; calibration then aligns the magnitudes with empirical frequencies so that one threshold transfers across splits. The shield consumes these calibrated variables as contracts instead of re-deriving risk from raw scores.

5.4. Shield Integration

The shield evaluates hard constraints on the final trajectory and, when available, symbolic contracts from NeSyWM. Each intervention logs the predicate, threshold, source, repair, fallback status, final command, and simulator outcome. This traceability is essential because the method comparison must distinguish NeSyWM itself from the external shield.

An external shield can repair commands only after receiving model scores or hand-coded state predicates. If a standard world model assigns low risk to a counterfactual candidate that is actually unsafe, the shield may inherit that false-safe interface. NeSyWM instead trains collision, off-road, lane, route, TTC, safe-stop, plausibility, and uncertainty variables as part of the latent state and then lets the shield audit those variables explicitly.

Ordinary predicate heads also leave a gap: they may be accurate on logged futures but poorly calibrated for planner-induced candidate actions. The model contribution is to make symbolic variables part of the training geometry through rule-energy, anchors, ranking, and calibration losses, so that latent rollouts are constrained by the same variables used for decision support.

6. Experimental Protocol

Experiments use MetaDrive because it supports procedural maps, traffic, and repeatable seeds (Li et al., 2021). The implemented stack logs structured ego state, route waypoints, nearby actors, event labels, TTC estimates, planner candidates, shield metadata, and optional front/top-down imagery. The visual training path stores 256 by 256 front-view RGB frames for training and 720 by 720 videos for audit evidence. The rasterized export contains 1,727 usable episodes after 13 skipped/corrupt episodes.

Counterfactual samples are split by seed/scenario, not by random frame. The CPU integration gate uses 3,863 train samples, 1,296 calibration samples, 1,327 test samples, and 702 hard-test samples at the primary 2 second horizon. The calibration split is used for temperature scaling and threshold fitting; test and hard-test are held out for reporting.

We report standard metrics such as latent loss, JEPA loss, future-feature loss, predicate AUROC/F1/precision/recall, and ECE, and functional metrics including false-safe rate, false-block rate, top-1 safe selection, top-k safe coverage, risk AUROC/AUPRC, lead time to danger, physical-plausibility violation, closed-loop collision/off-road/lane/TTC rates, longest off-road burst, first off-road step, off-road frames while shield status is safe, intervention rate, repair success, fallback rate, progress-collapse rate, latency, and approximate FPS.

The comparison families are BaseWM, BaseWM + Shield, NeSyWM, and NeSyWM + Shield. Ablations remove counterfactual data, symbolic losses, anchor alignment, triplet loss, calibration, latent rollout, predicate heads, action repair, route context, and the external shield. A PDM-style centerline/rule planner remains a strong closed-loop baseline because simple planners can be competitive (Dauner et al., 2023). For reproducibility, each orchestration script writes configs, seeds, scenario filters, checkpoints, commands, elapsed time, health state, and report paths; large datasets stay out of git, while report CSV/JSON files are deterministic.

7. Results

The results combine three evidence types. Tables 3–5 report CPU counterfactual experiments and family-level gates that are suitable for objective selection and benchmark validation. The visual/raster training paragraph reports checkpoint feasibility, and Table 6 reports an actor-aware raster counterfactual evaluation under fixed and calibrated thresholds. The deterministic scoring audit validates path resolution, health checks, confidence-interval generation, and the report schema used by the model-family comparison.

Rows are ranked for decision support rather than prediction fit alone: false-safe rate is prioritized first, followed by top-1 safe selection, risk calibration, and fi-

Table 1. Model families used throughout the paper. BaseWM and NeSyWM share the same visual/raster export and differ in whether symbolic risk state is internal to the world model.

Family	World model objective	Symbolic state internal?	Shield input
LegacyCPU-WM	early CPU prototype	no	ordinary predicates or none
BaseWM	latent/JEPA/future-feature + ordinary predicate heads	no	none
BaseWM + Shield	same as BaseWM	no	learned predicate heads + symbolic rules
NeSyWM	BaseWM + EBC, anchors, triplet, ranking, calibration	yes	none
NeSyWM + Shield	same as NeSyWM	yes	calibrated symbolic risk contracts

Table 2. Standard and functional metrics are kept separate because they answer different questions. This deterministic family-level scoring audit checks the reporting path for prediction fit versus candidate-decision utility.

Model	Latent ↓	AUC ↑	ECE ↓	False-safe ↓
BaseWM	0.16	0.82	0.08	0.2364
BaseWM + Sh.	0.16	0.82	0.08	0.2364
NeSyWM	0.18	0.84	0.05	0.1545
NeSyWM + Sh.	0.18	0.84	0.05	0.1273

nally standard loss. This ordering prevents a policy family from looking good by blocking almost everything, while still treating unsafe permission as the primary failure mode.

7.1. Functional Metrics Expose False-Safe Behavior

The counterfactual evidence bundle intentionally reports standard world-model metrics beside functional counterfactual metrics. The deterministic family-level scoring audit in Table 2 stresses the reporting path and the central evaluation claim: a model can have apparently strong latent/predicate columns while still having high false-safe counterfactual risk.

Latent loss and logged predicate AUC summarize how well a model matches the observed future distribution. Planning asks a different question: among the candidates not executed in the log, which one is safe enough to choose? A model can therefore look acceptable under standard columns while still ranking unsafe lateral or speed-modified candidates as safe. Figure 4 visualizes this gap for the scoring audit.

The scoring audit contains 1,980 deterministic outcome rows. Bootstrap intervals over candidate-tag rows give false-safe rates of 0.2364 [0.2015, 0.2682] for BaseWM, 0.1545 [0.1273, 0.1818] for NeSyWM, and 0.1273 [0.1030, 0.1515] for NeSyWM + Shield.

7.2. Rule-Energy Objectives Change Decision Behavior

The rule-energy ablation isolates the neuro-symbolic training mechanism. The positive-only variant reaches a near-zero validation loss but retains a 0.279 false-safe rate. Adding energy-based constraints and anchors reduces false-safe behavior below 1% on that validation split; adding triplet constraints improves risk ranking.

Table 3. CPU validation of rule-energy training. Low validation loss alone is not sufficient: the positive-only objective is the worst by false-safe rate.

Variant	Val loss ↓	FS ↓	AUC ↑	Anchor ↑
Positive only	0.0007	0.2790	0.5231	0.7210
EBC	0.2944	0.1614	0.7843	0.7320
EBC + anchor	0.2965	0.0075	0.7626	0.3285
EBC + anchor + triplet	0.3596	0.0081	0.8009	0.4484

Table 4. Calibrated CPU counterfactual metrics. Utility is lower-is-better in the reported implementation.

Split	Variant	Utility	FS	FB	AUC
Test	Positive	0.2466	0.0098	0.7159	0.4992
Test	EBC	0.1821	0.0158	0.5275	0.7499
Test	EBC+anchor	0.1692	0.0166	0.4559	0.7986
Test	EBC+anchor+tri	0.1554	0.0136	0.4137	0.8092
Hard	Positive	0.1688	0.0356	0.3917	0.4522
Hard	EBC+anchor	0.1342	0.0484	0.2977	0.6134

7.3. Calibration and Decision Utility

The calibration stage fits temperature scaling and conformal-style thresholds on a calibration split and evaluates on test and hard-test splits. A trivial conservative threshold can reduce false-safe errors by blocking many safe candidates, so we report a decision-utility score that penalizes false-safe, false-block, poor ECE, and poor ranking. On the normal test split, EBC+anchor+triplet gives the best utility among the tested variants. On the hard-test split, EBC+anchor has the best utility, while EBC+anchor+triplet keeps stronger test-split ranking.

7.4. Formal CPU Integration Gate

The CPU gate trains compact checkpoints for the four paper families and writes the exact report schema used by the visual/raster stage. On the normal test split, NeSyWM + Shield is best by decision utility. On hard-test, NeSyWM has the best utility, while NeSyWM + Shield reduces false-block relative to BaseWM families at higher AUC.

7.5. Training Feasibility Evidence

The visual/raster GPU run completed the training stage on the raster export. The export contained 1,727 usable raster episodes from 1,732 uploaded v1 episodes; the exporter reported 1,719 newly exported episodes, 13 skipped episodes, and 0 failures. Both

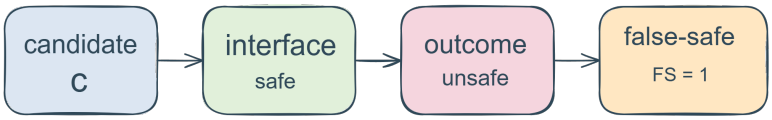

Figure 3. False-safe definition. A candidate is false-safe when the decision interface permits it as safe but the short-horizon counterfactual outcome label is unsafe.

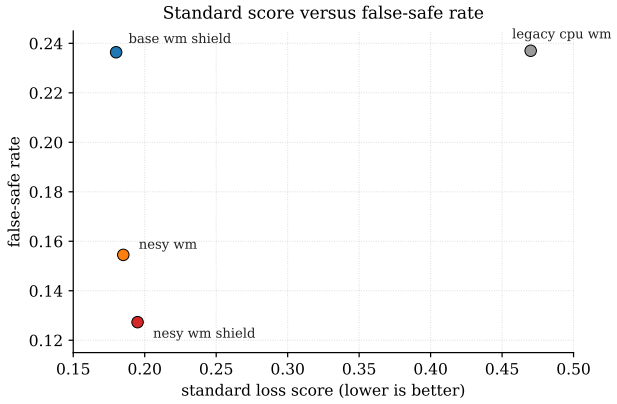

Figure 4. Standard world-model score and false-safe rate answer different questions. This motivates reporting functional candidate-decision metrics alongside latent and predicate metrics.

Table 5. CPU family comparison on the counterfactual evaluation gate. Utility is lower-is-better; FS and FB denote false-safe and false-block rates.

Split	Family	Utility	FS	FB	AUC
Test	BaseWM	0.2220	0.0000	0.7430	0.5846
Test	BaseWM+Sh.	0.1626	0.0075	0.5041	0.7818
Test	NeSyWM	0.1923	0.0030	0.6918	0.8141
Test	NeSyWM+Sh.	0.1554	0.0151	0.4017	0.8141
Hard	BaseWM	0.1650	0.0000	0.4060	0.3168
Hard	NeSyWM	0.1350	0.0199	0.3960	0.6036
Hard	NeSyWM+Sh.	0.1368	0.0442	0.3134	0.6036

trained model directories were stored locally and their PyTorch state dictionaries were loaded on CPU.

The BaseWM run trained over 194,470 windows with 4,944 validation windows; its loss moved from 1.6132 to 0.5945 over 10 epochs, with validation loss ending at 0.9851. The NeSyWM run used the energy-based anchor-triplet objective for 50 epochs on the same window counts; its logged training loss decreased from 9.6737 to 0.4765 with 7,188 rule samples. Both state dictionaries loaded after transfer: 48 tensors and 808k parameters for BaseWM, and 107 tensors and 1.278M parameters for NeSyWM. A 4-episode CPU closed-loop sanity evaluation completed in 79.0 seconds, observed no collision/off-road/lane-departure events, and logged latency, progress, and shield-contradiction fields with the same schema used by longer runs.

Table 6. Actor-aware raster counterfactual evaluation (2,100 held-out outcomes from 100 episodes). FS and FB are false-safe and false-block rates. The finding is diagnostic: on this larger sample the unshielded raster checkpoints rank candidates near chance, NeSyWM holds no advantage over BaseWM, and false-safe falls only under the shield, for both models, at a false-block cost.

Family	Threshold	FS ↓	FB ↓	AUC ↑
BaseWM	fixed 0.5	0.132	0.000	0.593
BaseWM	calibrated 0.045	0.125	0.029	0.593
NeSyWM	fixed 0.5	0.131	0.000	0.492
NeSyWM	calibrated 0.141	0.130	0.000	0.492
BaseWM+Sh.	fixed 0.5	0.003	0.078	0.982
BaseWM+Sh.	calibrated 0.85	0.004	0.023	0.982
NeSyWM+Sh.	fixed 0.5	0.000	0.292	0.987
NeSyWM+Sh.	calibrated 0.90	0.004	0.023	0.987

7.6. Actor-Aware Raster Counterfactual Evaluation

We evaluate the pulled visual/raster BaseWM and NeSyWM checkpoints on a bounded actor-aware CF-Safety subset (100 episodes, 3,228 outcomes at the 2 s horizon; 1,128 calibration and 2,100 evaluation outcomes). Queries now retain nearby-actor state, fixing an earlier limitation in which raster queries carried only ego and route context. We report both a fixed 0.5 decision threshold and a calibration-split threshold (Table 6).

We read this as diagnostic rather than as a superiority claim. On the larger sample neither unshielded raster model is a strong counterfactual critic: BaseWM reaches AUC 0.593 and NeSyWM only 0.492, close to chance, and both sit near a 0.13 false-safe rate at the fixed threshold. An earlier, much smaller subset had hinted at a slight ranking edge for NeSyWM, but it disappears once the sample grows. The safety, in every case, comes from the shield: after calibration both shielded families fall to near-zero false-safe at a false-block cost of about 0.023, and NeSyWM+Shield is if anything worse calibrated than BaseWM+Shield (ECE 0.389 against 0.120). Since the two checkpoints were trained on very unequal budgets in the submitted run (10 epochs for BaseWM, 50 for NeSyWM), we treat this as preliminary and revisit it under matched budgets below.

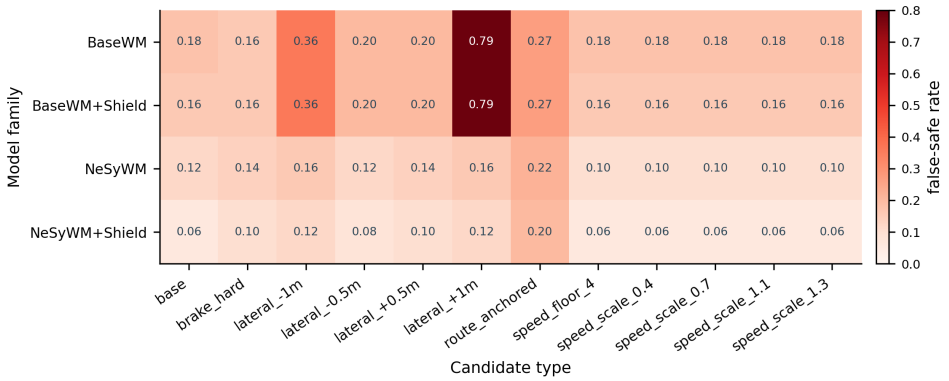

Figure 5. Candidate-type false-safe rate by model family. Lateral offsets and route-modified candidates are prominent sources of counterfactual pressure, which is why evaluating only logged futures is insufficient for planning.

7.7. External Sanity Check

To probe portability beyond MetaDrive, we confirmed that a lightweight headless driving environment (highway-env) installs and runs in our setup, and we exercised the family pipeline on a controlled synthetic stress set of 2,400 candidates (720 held out) under the same risk-aggregation and shield logic. We stress that this is a sanity check on the decision pipeline, not a rollout-based benchmark and not a numeric comparison against Think2Drive, AD-R1, or published PDM results. Even here the pattern recurs: the rule baseline and the shielded families keep false-safe low (0.032) while unshielded NeSyWM stays high (0.231), echoing the MetaDrive result that ranking gains become safe decisions only once a threshold or shield is applied. Full highway-env rollouts remain future work.

7.8. Epoch-Matched Raster Retraining

The submitted run trained BaseWM for only 10 epochs against NeSyWM's 50, so the two were never strictly comparable. To close that gap we retrained both for 50 epochs on the same 500-episode raster subset (38,719 training and 960 validation windows) on a single RTX 4090. At parity their validation losses are essentially equal, 0.089 for BaseWM and 0.094 for NeSyWM (Table 7), so the symbolic objective costs nothing in predictive fit. Re-running the actor-aware evaluation on the matched checkpoints gives the same picture as before: the unshielded models share an identical false-safe rate of 0.132, NeSyWM ranks candidates no better than BaseWM (AUC 0.575 vs. 0.424), and only the shield brings false-safe down to near zero (0.003 and 0.000). Matching the budget therefore leaves our conclusion intact: at raster scale it is the shield, not the unshielded model, that delivers safety.

Table 7. Epoch-matched raster retraining (both families, 50 epochs, identical 500-episode subset, single RTX 4090). Training losses are final-epoch values; raster FS is the fixed-threshold false-safe rate on a 100-episode actor-aware CF subset (2,100 held-out outcomes).

Family	Train loss ↓	Val loss ↓	Raster FS ↓
BaseWM (matched)	0.186	0.089	0.132
NeSyWM (matched)	0.455	0.094	0.132
BaseWM+Sh.	—	—	0.003
NeSyWM+Sh.	—	—	0.000

8. Discussion

The CPU evidence supports three claims: false-safe counterfactual metrics are separable from standard world-model metrics; positive-only training can look strong by loss while remaining weak as a safety critic; and rule-energy, anchors, triplets, and calibration improve functional utility in the integration gate. A shield bolted onto a standard world model can still inherit false-safe scores, so NeSyWM makes collision, off-road, route, TTC, safe-stop, plausibility, and uncertainty variables explicit in the latent state consumed by symbolic safety logic.

Attribution matters here. Across both the CPU gate and the raster evaluation, the component that actually drives false-safe to zero is the shield rather than the learned model, and it pays for that with a measurable rise in false-blocks. What the world model contributes is sharper risk ranking and calibration, which lowers the false-block price the shield must pay for a given level of safety. Keeping those two contributions apart, instead of crediting the model with safety the shield enforced, is what makes the comparison trustworthy. It also accounts for the raster numbers, where the shield reaches near-zero false-safe even though neither unshielded model ranks candidates well.

9. Limitations

This work is simulator-first. MetaDrive provides controllability and procedural diversity, but it is not a real-world dataset. Exact simulator state restoration is not exposed by our interface, so CF-Safety uses replay-near labels; our deterministic-replay audit over 1,000 candidates puts their fidelity at 75.6% agreement, which bounds how far label-derived claims should be pushed. The actor-aware raster evaluation covers a bounded subset (2,100 held-out outcomes from 100 episodes) and is diagnostic rather than a large-scale closed-loop result, though the epoch-matched retrain in Table 7 shows the conclusion holds once the training budgets are equalized. The external check uses a synthetic kinematic stress set rather than full highway-env rollouts. Broader validation should add exact-state or stronger re-simulation labels, more seeds and maps, PDM-style closed-loop baselines, larger raster evaluations, and progress-collapse ablations. We mitigate current threats by keeping standard and functional metrics separate, reporting false-block tradeoffs alongside false-safe, and separating model effects from shield effects throughout.

10. Conclusion

We presented a framework for evaluating and training driving world models as counterfactual safety critics. NeSyWM embeds symbolic risk variables into the latent model and exposes them to a traceable shield, while CF-Safety measures whether candidate actions are falsely permitted as safe. The evidence supports the paper's central claim: world models for safety-critical planning should be judged by functional decision utility, not only by logged future prediction.

References

Chen, D., Zhou, B., Koltun, V., and Krähenbühl, P. Learning by cheating. In Proceedings of the Conference on Robot Learning, volume 100 of Proceedings of Machine Learning Research, pp. 66–75. PMLR, 2020. URL https://proceedings.mlr.press/v100/chen20a.html.

Dauner, D., Hallgarten, M., Geiger, A., and Chitta, K. Parting with misconceptions about learning-based vehicle motion planning. arXiv preprint arXiv:2306.07962, 2023. URL https://arxiv.org/abs/2306.07962.

Ha, D. and Schmidhuber, J. World models. arXiv preprint arXiv:1803.10122, 2018. URL https://arxiv.org/abs/1803.10122.

Hong, Y., Zhou, X., Li, Y., Zhou, X., Liu, L., Luo, Y., Xu, S., Yang, L., and Song, Z. Drivefuture: Future-aware latent world models for autonomous driving. arXiv preprint arXiv:2605.09701, 2026. URL https://arxiv.org/abs/2605.09701.

Li, Q., Peng, Z., Feng, L., Zhang, Q., Xue, Z., and Zhou, B. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. arXiv preprint arXiv:2109.12674, 2021. URL https://arxiv.org/abs/2109.12674.

Li, Q., Jia, X., Wang, S., and Yan, J. Think2drive: Efficient reinforcement learning by thinking in latent world model for quasi-realistic autonomous driving (in carla-v2). arXiv preprint arXiv:2402.16720, 2024. URL https://arxiv.org/abs/2402.16720.

Maes, L., Le Lidec, Q., Scieur, D., LeCun, Y., and Balestriero, R. Leworldmodel: Stable end-to-end joint-embedding predictive architecture from pixels. arXiv preprint arXiv:2603.19312, 2026. URL https://arxiv.org/abs/2603.19312.

Sun, J., Sun, H., Han, T., and Zhou, B. Neuro-symbolic program search for autonomous driving decision module design. In Proceedings of the Conference on Robot Learning, 2020. URL https://corlconf.github.io/corl2020/paper_16/.

Yan, T. et al. Ad-r1: Closed-loop reinforcement learning for end-to-end autonomous driving with impartial world models. arXiv preprint arXiv:2511.20325, 2025. URL https://arxiv.org/abs/2511.20325.