Meadow: A Physics-Grounded Three-Layer Learning Stack for Robot Control

Sheng-Kai Huang

Independent Researcher, Taipei, Taiwan · *akai@fawstudio.com*

Technical Report · Version 0.0.2 · April 2026

Paper Code

ABSTRACT

We present *Meadow*, a learning stack for robot control whose inductive bias is grounded in physical state rather than perceptual observation. Meadow factors the control problem into three explicit layers: (i) a physics layer that expands a causal tree of candidate trajectories under simulator dynamics, (ii) a neural scorer that learns success likelihood over the resulting branches, and (iii) a neural reaction policy that distills the selected chain into a deployable low-latency controller. A named-variable calibration loop closes the system: when deployed behavior diverges from the predicted chain, specific physical parameters in the spec are re-estimated and the affected layer is re-distilled locally, without full retraining or demonstration re-collection. We report end-to-end results on four representative tasks (TwoRoom, PushT, Reacher, OGBench Cube) with per-task scorer validation accuracy up to 0.9990, sub-pixel reaction precision on PushT (0.81 px final block error), and a single-pipeline pass through four OGBench v1 families using 443–2,724 trajectory samples per family. Reaction inference is 0.095–0.122 ms on Apple Silicon (Core ML).

PREFACE — ORIGIN

Physics is what I love most, and it is what brought me here. At university, the Copenhagen interpretation rearranged how I see the world — wave–particle duality, the uncertainty principle, the probability wave, and its collapse. Years later, when AMI Labs articulated the framing — **intelligence does not begin in language but in the world** *— it landed for me with the same force.*

In my view, it is the same conviction restated in a different field: an intelligent agent and the world in which it acts are not separable, and the structure of intelligence reflects the structure of physical interaction. I found that vision deeply compelling, and Meadow is, in part, my own attempt to take it seriously.

Meadow tries to start the model on the side of physical state. This choice is informed by an observation from developmental neuroscience: an infant's first contact with the world does not seem to be visual, but rather vestibular, tactile, auditory, and respiratory; visual capacity arrives later, and the world is then re-understood through repeated physical interaction with objects. Meadow's first layer accepts these physical channels as first-class observables — the system begins by acting on physics, doing whatever it can in the world, while attending to the causal relationship between its outputs and inputs and, through the strength of those causal links, gradually composing multiple chains for exploration and task completion. The second layer takes these chains, together with the strongest causal chain for the task, as training data, with the aim of forming an understanding of abstract structure. The third layer is the trained policy itself, kept in living contact with the first layer's causal chains and corrected in real time whenever a gap appears. The hope is to remain consistent in spirit with the position argued in **A Path Towards Autonomous Machine Intelligence** *[1].*

I submit this report in homage to those who came before. If there are citation mistakes or misunderstandings on my part, I welcome guidance and correction.

CONTENTS

1. Introduction

The dominant paradigm in modern robot learning treats perception as the primary substrate: a high-capacity neural network ingests pixel observations and emits actions, with the burden of recovering physical structure left to gradient descent over large datasets [1]. This approach has produced impressive results, but couples two distinct learning problems — perceptual representation and control — into a single optimization, and inherits the well-known ill-posedness of inferring physical quantities (mass, friction, contact) from monocular pixel sequences.

An alternative starting point is suggested by general consensus in developmental neuroscience: vestibular, proprioceptive, and tactile signals develop in utero and are functionally mature at birth, while visual cortex requires months of postnatal development to reach adult performance. The first models a biological agent forms of its own physical world are constructed from physical, not perceptual, primitives.

We take this observation seriously as a design principle. Meadow accepts physical state as a first-class observable: device parameters, contact forces, joint positions, and goal thresholds enter the system as typed variables in a declarative spec, not as quantities to be recovered from images. The system then performs causal expansion over this state under a physics backend, learns to value branches with a neural scorer, and distills the resulting chain into a compact reaction policy. We show that this factoring yields three properties simultaneously: (i) explicit and named inductive bias at every layer, (ii) sample-efficient learning under small training corpora, and (iii) a calibration loop that, on deployment drift, identifies the specific physical parameter that changed and re-distills locally.

Contributions.

- A three-layer architecture that separates physics expansion, value learning, and policy distillation into independently inspectable modules (§2).
- A named-variable calibration loop that operationalizes drift detection and correction over the spec's physical variables, in contrast to opaque "concept drift" detection (§3).

- End-to-end empirical results on four task families with reproducible artifacts (§4), including sub-pixel reaction precision on contact manipulation and a single-pipeline pass through four OGBench v1 families.

2. Architecture

Meadow factors robot control into three layers, each with an explicit role and an explicit (or explicitly absent) learner. Inputs to the system are three typed specifications: a `DeviceSpec` declaring actuators, joints, and physical constraints; a `SceneSpec` declaring objects and contact geometry; and a `GoalSpec` declaring task thresholds and safety bounds.

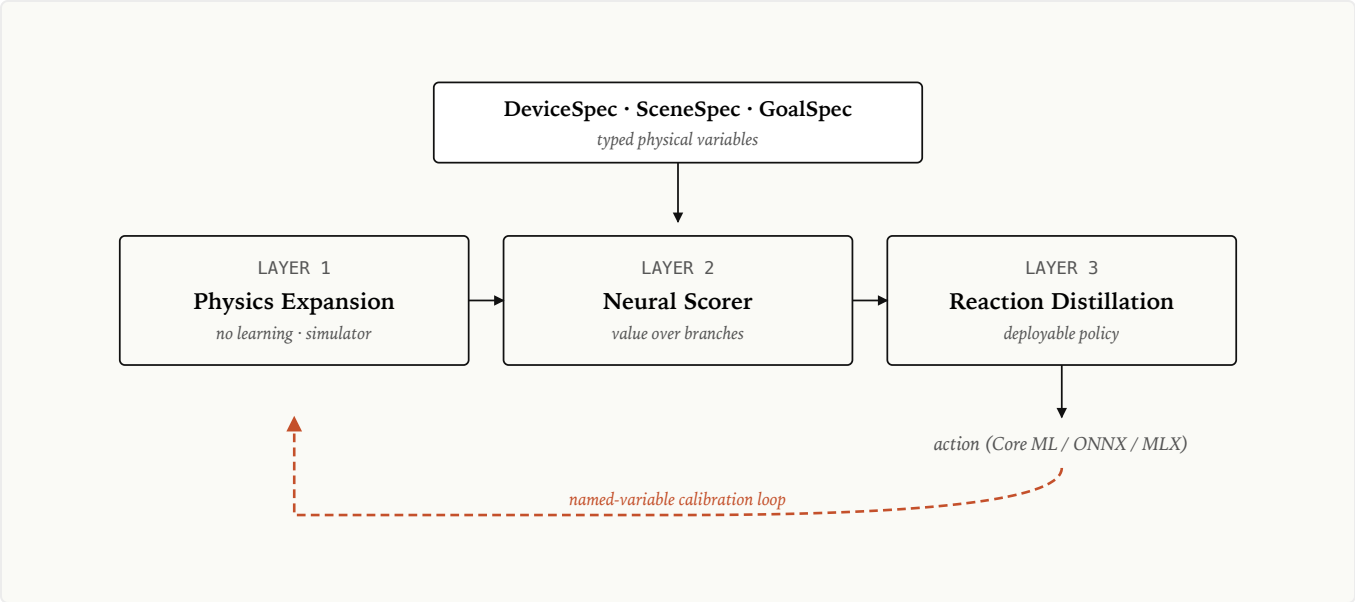


Figure 1. *The Meadow stack. Physics expansion is non-learning; the two neural layers receive a training distribution that has already been filtered through physical constraint, so the abstractions they form respect conservation and feasibility without auxiliary losses. The orange dashed path is the post-deployment named-variable calibration feedback.*

2.1 Physics Expansion (Layer 1)

Given a spec and an initial state, the simulator advances candidate futures under physical dynamics. Branches are pruned when they violate physical constraints encoded in the spec (joint limits, collision, control saturation) or task constraints encoded in `GoalSpec`. The result is a tree whose nodes are physically reachable states and whose edges are admissible

transitions. We retain successful chains, near misses, and pruned failures as structured records.

2.2 **Neural Scorer (Layer 2)**

A neural network is trained to map (state, candidate continuation) pairs to a continuous success score. Crucially, the training distribution is the physics-filtered tree from Layer 1: positive examples are successful chains, negative examples are pruned or near-miss branches. No separate hard-negative mining is required, and the scorer's learned representation inherits physical consistency from the substrate.

2.3 **Reaction Policy (Layer 3)**

The reaction policy is a compact neural network distilled from the chain selected by the scorer. We use behavior-cloning-style or k-nearest-neighbor exemplar variants depending on task structure (see §4). The policy operates over the same physical state space declared in the spec, so its inputs and outputs remain semantically inspectable.

> *Note that the spec layer is observation-source agnostic. Where simulator state is unavailable, the spec frame can be populated by a learned encoder — visual, tactile, multi-modal. Layers 1–3 are unchanged. We discuss this integration point in §5.2.*

3. **Calibration Loop**

A central property of the spec-driven design is that all variables relevant to learning have *names*: friction coefficient, payload mass, damping ratio, sensor offset. When a deployed reaction policy produces trajectories that diverge from the predicted causal chain, the deviation can be attributed not to opaque "concept drift" but to identifiable physical parameters whose values have moved.

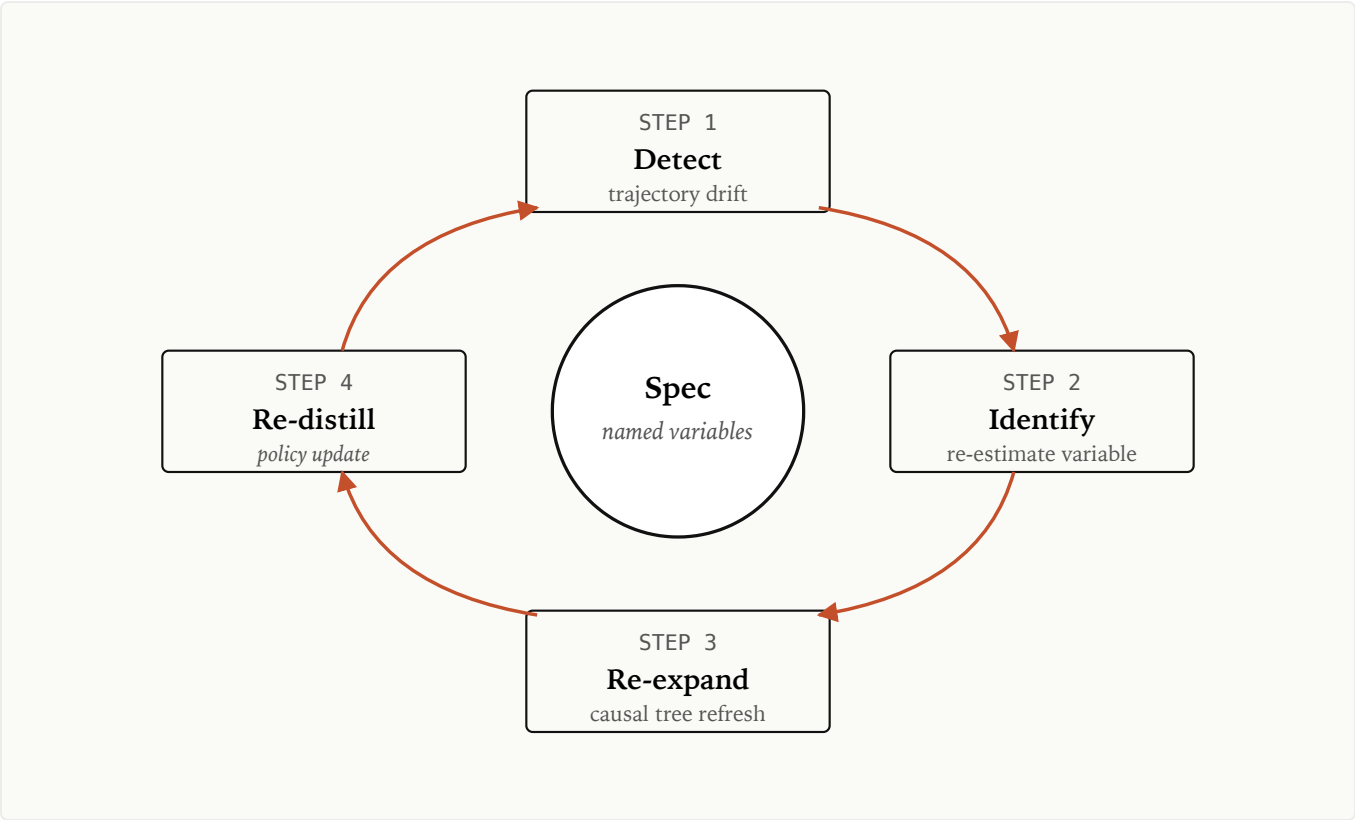


Figure 2. *The calibration loop. Drift is observed at the trajectory level, attributed at the parameter level via named-variable system identification, and corrected at the policy level via local re-distillation. The central spec is the single source of truth for all named physical variables; prior policy versions are retained for rollback.*

This contrasts with end-to-end learned controllers, where drift requires either (i) the collection of new demonstrations under the changed conditions, or (ii) full retraining of an opaque policy whose internal representation cannot be inspected for the offending dimension.

4. Experiments

4.1 Showcase Tasks

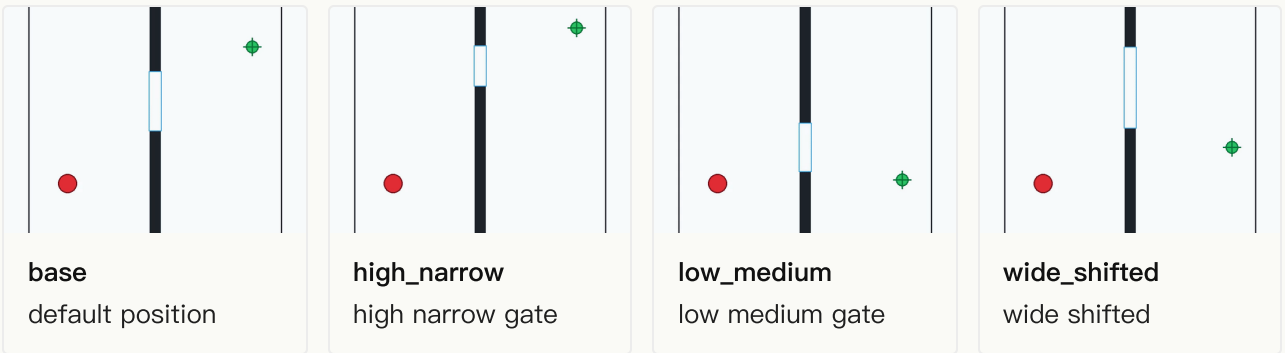
We instantiate the Meadow stack on four tasks spanning navigation, contact manipulation, arm control, and pick-and-place. Table 1 reports per-task scorer and reaction performance; Figure 3 shows representative rollouts. All numbers are from local artifacts; no claims are extrapolated to public benchmarks.

Task	Causal / Scorer	Success Rate	Precision	Train Samples	Device	Iter
TwoRoom (2D nav)	val_acc 0.9990 (79/80)	80/80 (100%)	binary terminal	49,920	M1 Max · MLX	1,799
PushT (contact)	4/4 pose-aware guard	2/2 (100%)	0.81 px final block	1,010 / 315	M1 Max · MLX	— (oracle)
Reacher (arm)	4/4; median 0.00369	2/2 (100%)	mean final dist 0.00494	288 / 144	M1 Max · MLX	— (oracle)
OGBench Cube	val_acc 0.9963 (60/60)	4/4 (100%)	phase-conditioned	35,200	M1 Max · MLX	1,399

Table 1. *End-to-end results on four representative tasks. The Causal / Scorer column reports causal-tree selection succe applicable, scorer validation accuracy. The Success Rate column reports the distilled reaction policy's heldout success ra column reports task-specific fine-grained metrics. Train Samples: scorer-side single count; reaction-side as train / val. D platform (Apple Silicon GPU via MLX, 64GB unified memory). Iter is scorer training steps; PushT and Reacher use an causal-tree planner instead of a neural scorer. Wall-clock is measured on the device, reconstructed from local session ti checkpoint mtimes; ¹ PushT oracle 4-episode recording, ² PushT student v10 reaction distillation, ³ Reacher recording w separate session — exact wall-clock pending. Detailed per-checkpoint scorer convergence is captured in the local* `metr` *(available on request).*

TwoRoom · 2D navigation

CAUSAL CHAIN · NEURAL SCORER ROLLOUTS (4 REAL SCENARIOS FROM THE VAL_ACC 0.9990 TRAINING CHECKPOINT)



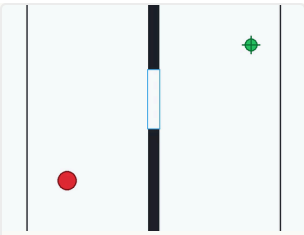
base
default position

high_narrow
high narrow gate

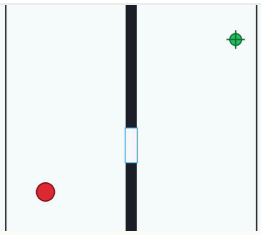
low_medium
low medium gate

wide_shifted
wide shifted

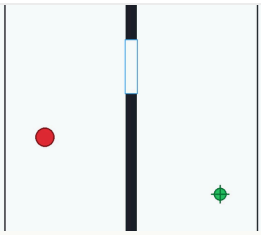
REACTION POLICY · BC STUDENT (4 SUCCESSFUL SCENARIOS)



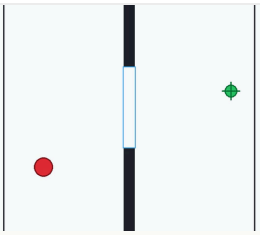
base
default position



high_narrow
high narrow gate



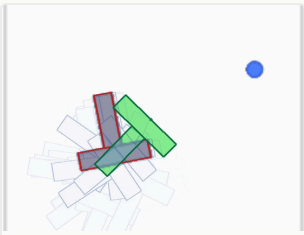
low_goal
low goal



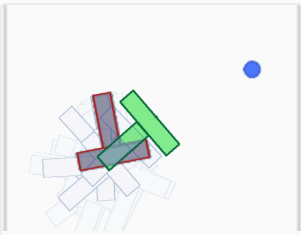
wide
wide scene

PushT · contact manipulation

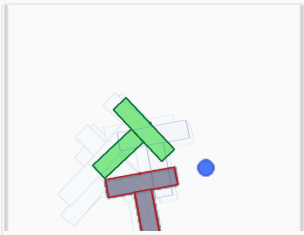
CAUSAL TREE · TRUE-ENV CAUSAL TREE (4 SUCCESSFUL SCENARIOS)



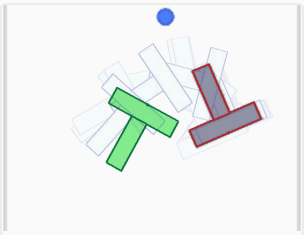
base
default push



long_push
long-distance push



offset
offset start

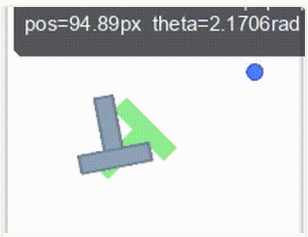


rotation
rotation align

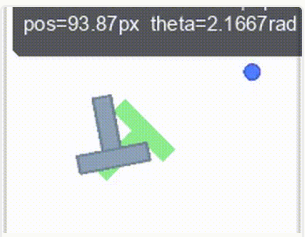
REACTION POLICY · KNN EXEMPLAR REACTION V10 (4 SUCCESSFUL ROLLOUTS)

episode 4
final 0.81 px

episode 17
final 2.32 px



episode 0
final 1.43 px



episode 18
final 1.53 px

Reacher · arm control

CAUSAL TREE · TRUE-ENV CAUSAL TREE (4 SUCCESSFUL SCENARIOS)

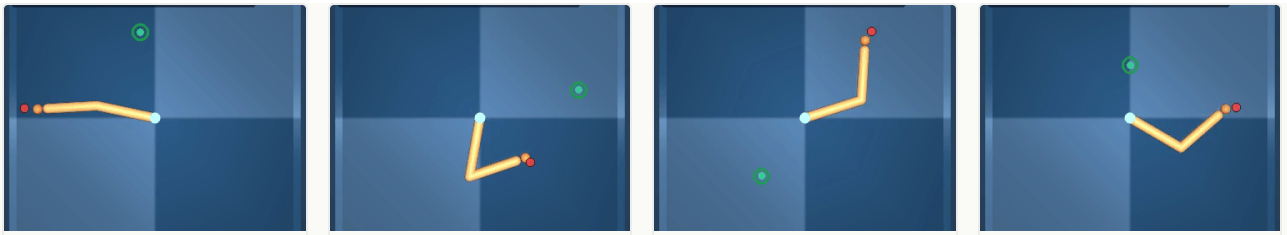
base
seed 0, final 0.0031

offset
seed 1, final 0.0036

wide
seed 4, final 0.0040

hard
seed 7, final 0.0038

REACTION POLICY · REACTION V2 (4 SUCCESSFUL ROLLOUTS)



val seed 4
final 0.0060

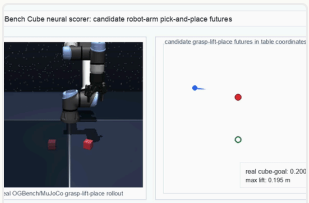
val seed 5
final 0.0039

train seed 0
final 0.0066

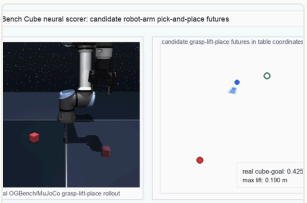
train seed 1
final 0.0064

OGBench Cube · pick-and-place

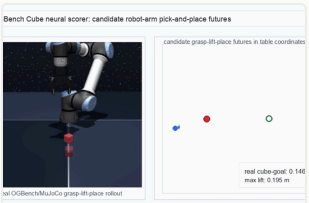
CAUSAL CHAIN · NEURAL CAUSAL SCORER ROLLOUTS (4 SUCCESSFUL SCENARIOS)



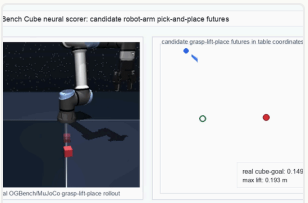
base
default start



hard_start
hard start

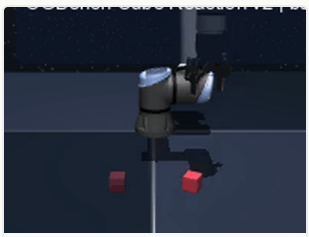


high_goal
high goal

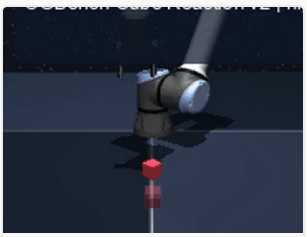


side_goal
side goal

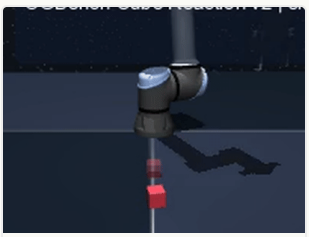
REACTION POLICY · REACTION V2 (4 SUCCESSFUL ROLLOUTS)



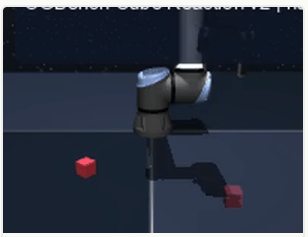
base
final 0.0130



high_goal
final 0.0037



side_goal
final 0.0031



hard_start
final 0.0042

Figure 3. *Successful rollouts across the four tasks, showing both the causal chain (true-env causal tree teacher) and the reaction policy (distilled student) completing successfully across varied starting positions and goal positions. All videos are loaded directly from local artifacts; PushT, Reacher, and OGBench Cube reaction rollouts were re-recorded locally and integrated into this page.*

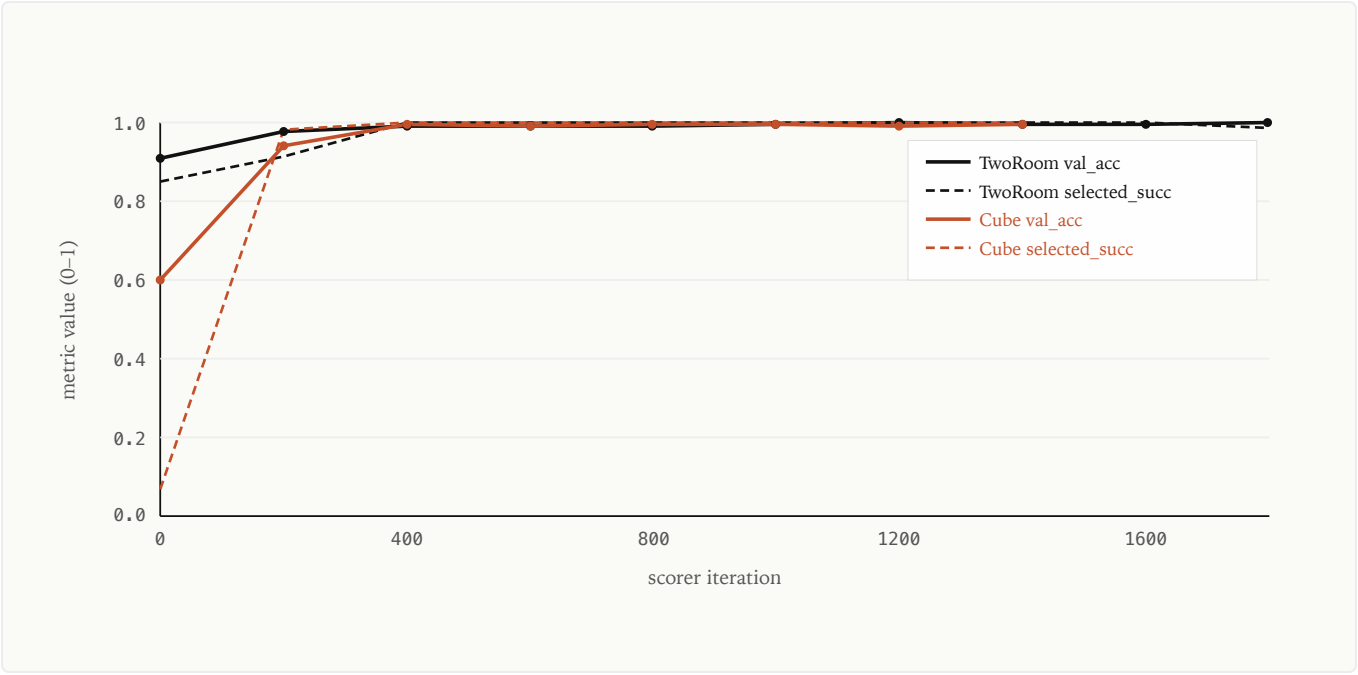


Figure 5. *Neural scorer training convergence on TwoRoom and OGBench Cube, sourced from local* `metrics.json`*. Solid lines: val_acc; dashed lines: selected_success_rate. Cube's selected success rate jumps from 6.7% at iter 0 to 98.3% within 200 iters — demonstrating the sample efficiency of "physics-filtered training data + a compact scorer." Final val_acc: TwoRoom 0.9990 (iter 1799), Cube 0.9963 (iter 1399).*

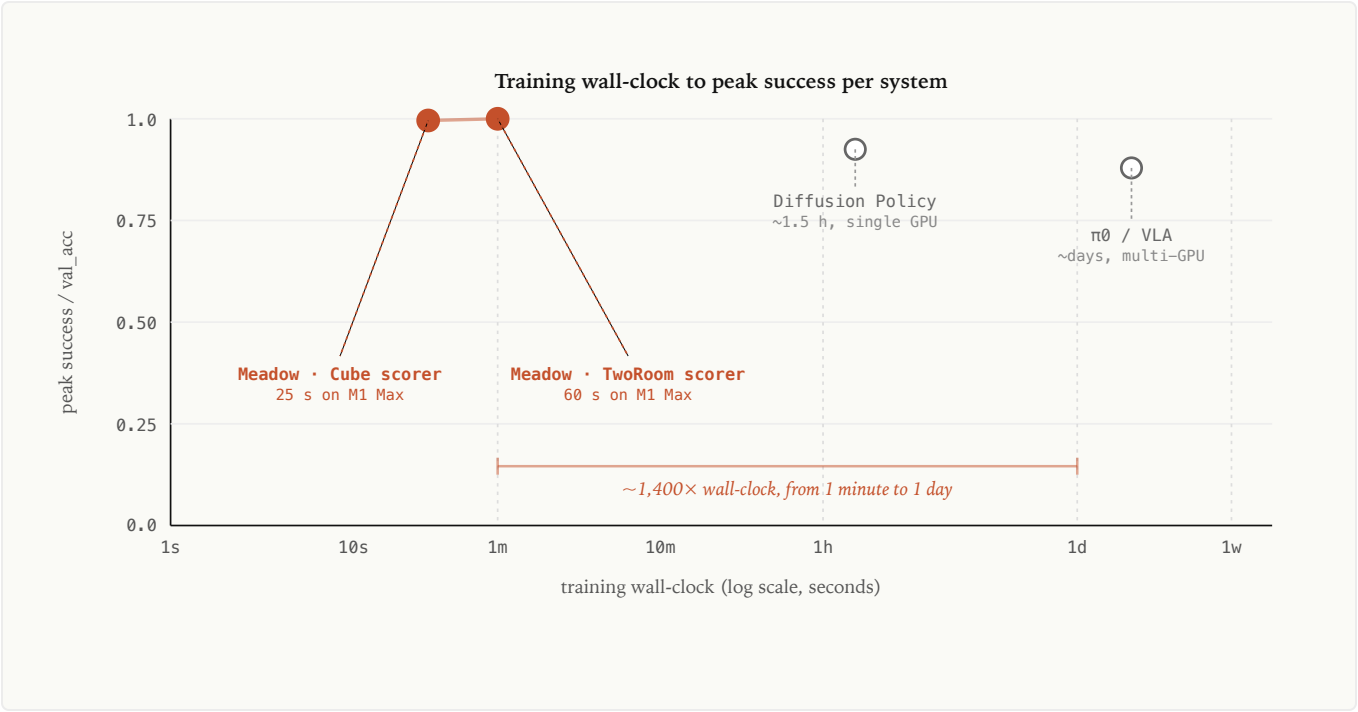


Figure 6. *Training wall-clock comparison. X-axis: log-scale training wall-clock (seconds); Y-axis: peak success rate / val_acc reached by each system.* ***Filled orange markers****: Meadow measured on Apple M1 Max + MLX (TwoRoom scorer 60 s · val_acc 0.999; Cube scorer 25 s · val_acc 0.996).* ***Hollow grey markers****: Diffusion Policy and π0 order-of-magnitude figures from public sources. Important caveat: industry sources rarely publish exact wall-clock (see Figure 7); hollow markers are reasonable estimates inferred from each project's hardware footprint and dataset scale, not a head-to-head benchmark. Task contracts differ — Meadow's scope is intentionally narrowed to spec-driven control, not general visuomotor policy; this figure does not claim ML capability*

superiority. It quantifies the consequence of a "device-side, no-cloud" deployment-surface choice — at comparable success rates, Meadow runs on a consumer-grade Apple Silicon laptop while the comparison group requires single-to-multi GPU.

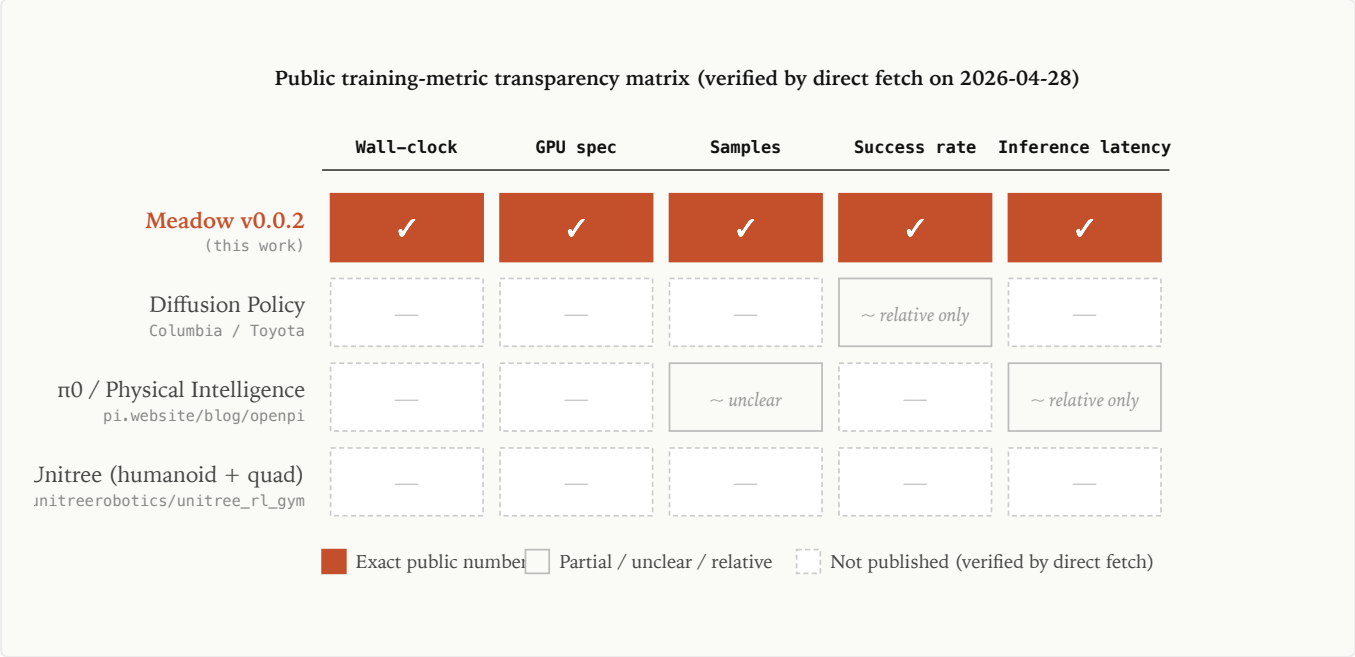


Figure 7. *Transparency matrix of training metrics across the robot-learning literature. Source: direct fetches of each project's public pages performed by this research on 2026-04-28. Meadow v0.0.2 publishes* ***exact numbers across all five key metrics*** *(wall-clock, GPU specification, sample scale, success rate, inference latency) — each verifiable on this page in Figure 5, Table 1, and Table 2. The three comparison systems publish limited information: only relative values (e.g., π0-FAST "4–5× higher inference cost") or qualitative descriptions, and* ***none publishes the full wall-clock + GPU + samples + success rate + latency tuple****. This figure does not accuse any system of being "opaque": business reasons and trade secrets are reasonable non-disclosure motives. The figure simply makes Meadow's radical-reproducibility choice explicit — for academic citation, independent verification, and customer due diligence, this density gap is materially usable.*

4.2 OGBench v1 Single-Pipeline Pass

To test whether the same expansion–scorer–reaction pipeline transfers across task families with only a spec change, we ran the OGBench v1 [2] API check across four families. Results are summarized in Table 2. The same code path produces 8/8 on every family with task-specific sample counts between 443 and 2,724.

Family	Environment	Causal	Reaction	Samples	Causal / React (ms)
Cube	`cube-single-singletask-task1-v0`	8/8	8/8	894	70.96 / 73.55

Family	Environment	Causal	Reaction	Samples	Causal / React (ms)
PointMaze	`pointmaze-medium-singletask-task1-v0`	8/8	8/8	2,724	21.99 / 22.74
Scene	`scene-singletask-task1-v0`	8/8	8/8	903	74.61 / 64.17
Puzzle	`puzzle-3x3-singletask-task1-v0`	8/8	8/8	443	52.34 / 36.42

Table 2. *Single-pipeline pass over four OGBench v1 families. The expansion – scorer – reaction code path is identical across rows; only the spec changes. Latency columns report mean wall-clock per causal expansion and per reaction inference (uncompiled Python; deployed Core ML reaction is 0.095 – 0.122 ms).*

5. Discussion and Related Work

5.1 Relation to Pixel-based World Models

Joint-embedding predictive architectures [1] address a problem that Meadow does not: learning a predictive latent representation from raw pixel observations without representation collapse. We view our work as building from the other side of the same bridge. The spec layer in Meadow accepts physical state as input; where such state is unavailable, we anticipate that JEPA-class encoders or analogous learned encoders will populate the spec frame upstream of Layer 1. The two lines of work seem to us most naturally read as two layers of one eventual stack — perception on one side, named-physical planning and control on the other.

This factoring also separates two questions that are sometimes conflated. One is the open empirical question of whether predictive self-supervision *emerges* physically consistent representations from raw observation. The other is the engineering question of how to plan and control once one has physically named state. Meadow takes no position on the former. It offers an explicit, inspectable structure for the latter, in the hope that the two answers can eventually meet in the middle.

One choice is worth stating directly because it is sometimes mistaken for a return to hand-crafted priors: *Meadow's spec is an explicit, named description of the body, joints, contact geometry, and goal — not implicit, but no greater in human-effort cost than what already happens in any robot learning pipeline today.* Diffusion Policy inherits the same physical

specification through teleoperated demonstrations on a specific robot; π0 inherits it through multi-robot dataset assembly; JEPA-style world models [1] inherit it through their training environment; classical control inherits it through the URDF and MuJoCo / Isaac models that any simulation pipeline already requires. The Meadow choice is not to introduce more human-authored prior — it is to keep the description that *everyone already needs* as a first-class, named, inspectable input rather than burying it in dataset collection or simulator setup. The benefit is precisely what makes the calibration loop in §3 possible at all: when reality drifts, a named variable can be re-estimated; an implicit one cannot.

5.2 Multi-Modal Grounding

The spec interface is typed but observation-source agnostic. Proprioceptive joint encoders, force/torque sensors, IMU streams, and tactile arrays can each enter as a typed channel without changing Layers 1–3. Integration with these modalities is an active engineering direction; we view it as a natural extension of the developmental-neuroscience starting point articulated in §1, where vestibular and proprioceptive signals predate vision.

5.3 Limitations

- **Simulator dependence.** Layer 1 currently relies on an external physics simulator (e.g., MuJoCo). Sim-to-real gap is therefore inherited from the chosen backend. The calibration loop partially mitigates this by re-estimating spec parameters from observed deployment behavior, but extreme regimes (deformable bodies, fluid contact, complex friction) remain bounded by simulator fidelity.
- **No pixel-only mode in current release.** A pixel encoder feeding the spec frame is on the roadmap (§5.2) but not in v0.0.2.
- **Calibration loop ergonomics.** The drift-detection and re-estimation cycle is operational on a single workstation today; productizing it as a turnkey on-device service for non-ML operators is engineering work still in progress.

5.4 Quantum-Inspired Structural Correspondences

This work originates from a few core ideas in the Copenhagen interpretation. In implementing the causal-tree architecture, we found that three of these concepts have natural structural correspondences with Meadow's design, one is a moderate structural

analogy, and two important structures — branch interference and entanglement — are not yet implemented. The latter constitute natural directions for future work.

Copenhagen concept	Correspondence in Meadow	Strength	Currently absent
Wave–particle duality	Think mode (branched distribution) ↔ Reaction mode (single distilled trajectory)	Strong	—
Probability wave (Ψ)	The whole causal tree ≈ distribution over possible futures; scorer score ≈ $\|\Psi\|^2$	Strong	**Branch interference:** branches are currently independent; no superposition term
Wavefunction collapse	Scorer-selected chain ≈ collapse from branched possibilities to a single trajectory (the architecture diagram literally names this stage `Collapse`)	Strong	—
Uncertainty principle	Conjugate trade-off between think-depth and reaction-latency; between spec resolution and compute cost	Moderate	No quantitative bound expressed as an inequality
(Derivative) Entanglement	—	—	**Coupling of causal trees** across multiple agents that share spec variables; not implemented

Table 3. *Structural correspondences between four concepts of the Copenhagen interpretation and the Meadow causal-tree architecture. The first three are strong correspondences; the uncertainty principle is a moderate structural analogy; entanglement is listed as a derived, not-yet-implemented item. The right column doubles as a roadmap.*

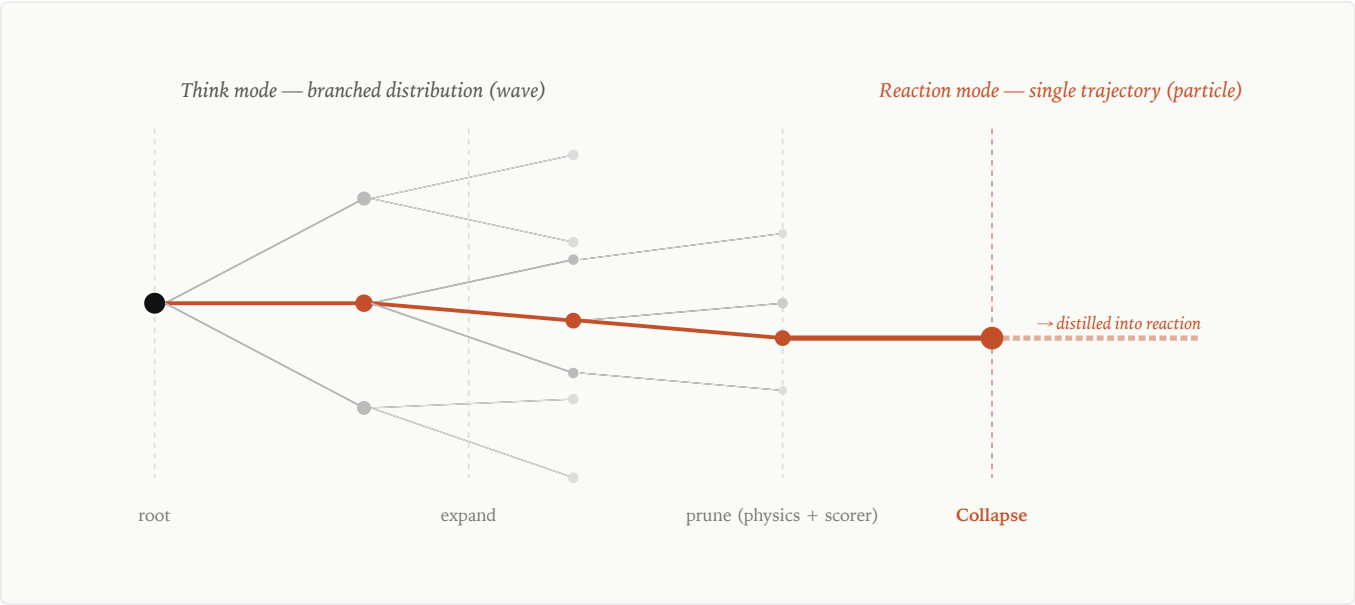


Figure 4. *Causal-tree expansion and collapse. Grey branches are candidate futures pruned by physical constraint or by the scorer; the red path is the single chain selected by the scorer that proceeds into reaction distillation. The same system presents as a "branched distribution" (wave) in Think mode and as a "single trajectory" (particle) in Reaction mode — a visual instance of the wave–particle and wavefunction-collapse correspondences listed in Table 3.*

The three currently absent structures suggest concrete research directions:

- **Branch interference.** Introducing a path-integral-inspired scorer that allows the "phases" of multiple trajectories to superpose — directly relevant to scenarios where two seemingly independent policies constructively or destructively interfere on certain states.
- **Multi-agent entanglement.** When multiple robots share spec variables (joint manipulation, coordination), their causal trees naturally entangle — this aligns with the longer-term roadmap toward collective coordination.
- **Quantitative conjugate bound.** Expressing the think-depth × reaction-latency trade-off as an optimizable inequality, allowing the system to find optimal allocation between conjugate quantities automatically.

6. References

[1] Y. LeCun. *A Path Towards Autonomous Machine Intelligence*. OpenReview, 2022.

[2] S. Park, K. Frans, B. Eysenbach, S. Levine. *OGBench: Benchmarking Offline Goal-Conditioned RL*.

arXiv:2410.20092, ICLR 2025.

Hey-Meadow Lab · github.com/Hey-Meadow/meadow-wm · akai@fawstudio.com