C-PHAST: Compositional Port-Hamiltonian World Models for Structured Dynamics Transfer

Shubham Bhardwaj [1 2] **Chandrajit Bajaj** [1]

Abstract

World models for robots and engineered systems should survive redeployment across the physical scales at which systems actually change: components, ports, contacts, graph edges, sensing channels, and candidate actions. A push on a robot may become joint work, contact loss, damping, or sensing degradation; a microgrid load change may become capacitor storage, line current, resistive loss, or voltage-regulation error; and an excitable-cell stimulus may become fast voltage, slow recovery, or diffusive spread. Fixed-dimensional dynamics models cannot be resized across such deployments, while unconstrained graph models can process variable layouts but may ignore how energy is stored, dissipated, and exchanged. We introduce **C-PHAST** (Compositional Port-Hamiltonian Architecture for Structured Temporal Dynamics), a compositional port-Hamiltonian world model that learns reusable physics-structured dynamics from observed rollouts by representing a deployed system as shared typed local physical blocks connected by power-preserving interconnects and explicit input ports. The same learned block library can therefore be re-instantiated under declared multiscale changes in subsystem count, robot embodiment metadata, excitable-circuit size, microgrid topology, or observation interface. At rollout time, C-PHAST exposes predictions as typed consequences, including energy use, contact/support demand, port work, residuals, sensing value, and stability margins, before a downstream planner or critic turns them into scores or vetoes. Under explicit assumptions we make these claims quantitative: a finite-step energy-balance defect, an observer-aware transfer bound, and typed-consequence decision margins, certified on known-operator deployments and empirically calibrated otherwise.

Across robot, excitable-circuit, and electrical-system benchmarks, each result tests a different part of this redeployment contract. Isaac Lab cross-robot transfer tests the rollout substrate: from ANYmal-D to Unitree Go2, 50-step open-loop error degrades only $1.6\times$, while recurrent baselines degrade by an order of magnitude and graph baselines collapse; when the same rollout is instead re-anchored in measurements every five steps, as the deployment contract prescribes for state tracking, C-PHAST also attains the lowest absolute transfer error of any model family tested. Coupled FitzHugh–Nagumo circuits test repeated-block recomposition outside robotics, transferring from $N{=}3$ cells to unseen $N \in \{2, 4, 5, 6, 8\}$ with essentially flat one-step error. DC microgrids test heterogeneous DGU/line composition: the specialized N-PHDAE baseline remains stronger in the full-state known-graph regime, but topology reconfiguration favors C-PHAST PARTIAL as a tighter design surrogate and sparse sensing remains competitive rather than collapsing. The decision-facing tests ask a different question: can the same forecast object compare possible next behaviors before an objective collapses them to a score? C-PHAST rolls each option forward and reports what it would spend, sense, stress, or risk; in Isaac Lab, this interface flags all failing lateral-push futures before simulator-level failure. On real Spot-with-arm logs from six teleoperated episodes and 1,070 five-second windows, the telemetry-to-action-card interface predicts the named consequences behind efficiency, safety, mapping, balance, and inspection cards; on 173 held-out windows it reaches mean Pearson/Spearman $0.880/0.860$, with leave-one-episode-out agreement $0.652/0.545$.

[1]Department of Computer Science, Oden Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, TX, USA [2]Mihawk.ai. Correspondence to: Shubham Bhardwaj <shubham.bhardwaj@utexas.edu>.

Preprint. July 28, 2026.

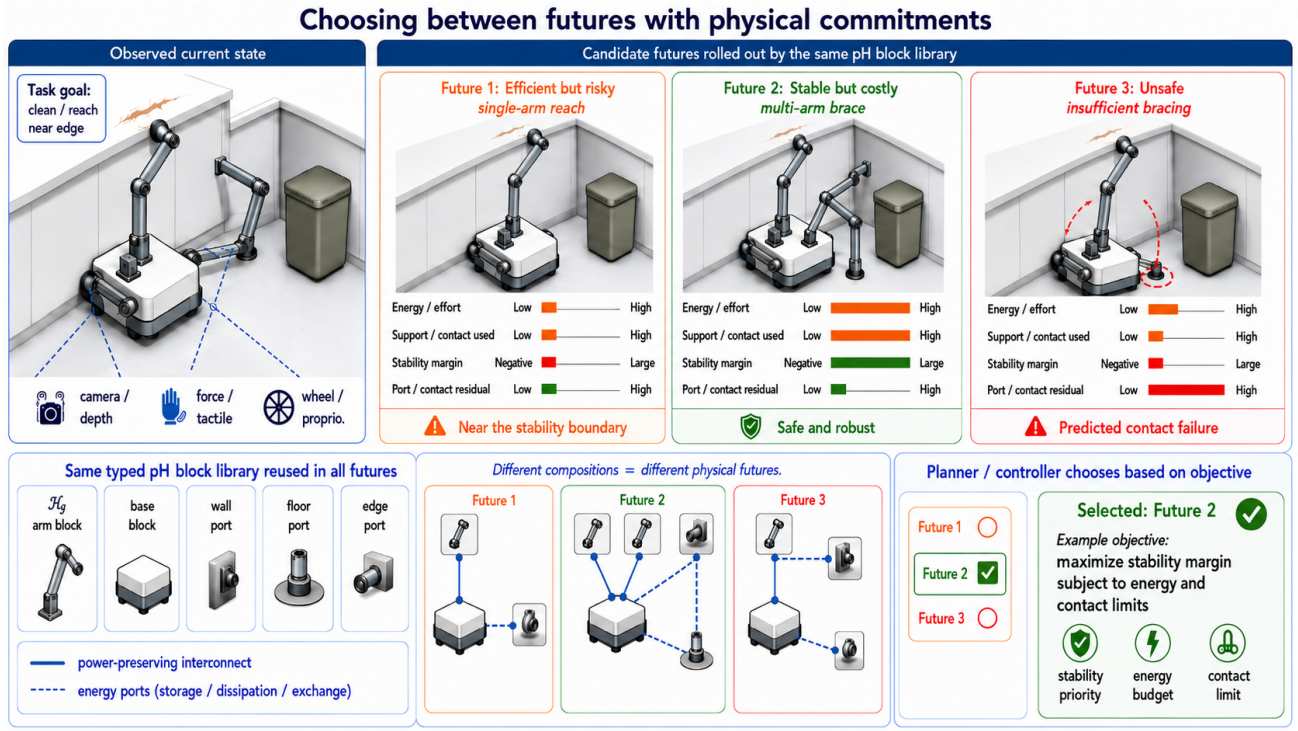

Figure 1. **Choosing between futures with physical commitments.** From the same observed state, C-PHAST rolls out candidate futures by reusing a typed pH block library under different active physical parts, contacts, and couplings. Each rollout exposes predicted energy use, support/contact allocation, stability margin, and port/contact residuals. The planner turns these typed readouts into objective-specific factors, so the same learned world model can support safety-first, energy-efficient, or task-progress choices without retraining local dynamics. Figure 4 formalizes this interface as a reusable typed-block rollout contract, and Figure 10 shows a concrete Isaac Lab branch from the same state.

1. Introduction

Learned world models are useful only if they can still be queried after deployment changes and before the system commits to a future. A robot does not execute inside one fixed dynamical graph: contacts switch, support roles change, arms or legs are reassigned, and candidate actions trade energy use against stability margin. An excitable-cell system likewise changes its effective composition when cells are added or diffusive links are rewired. An electrical network likewise changes its effective graph through feeder switching, distributed generation unit (DGU) addition or removal, and sensing loss. In all cases, the local component physics may be reusable, but the deployed composition that routes power through the system changes, and the decision layer needs to know what each possible future physically costs.

A useful way to view these settings is as disturbance-routing problems. Multiscale thermodynamic views argue that driven systems often respond by transporting or dissipating excess energy across several scales (Venegas-Aravena & Cordaro, 2024). We use only the operational version of that idea. A push on a robot may become body motion, joint work, contact loss, damping, or sensor degradation; a load change in a microgrid may become capacitor storage, line current, resistive loss, or voltage-regulation error; a stimulus in an excitable circuit may become a fast voltage excursion, slow recovery, or diffusive spread to neighboring cells. C-PHAST does not discover every physical scale automatically. It gives the deployed model declared typed storage, dissipation, ports, observations, and interconnections through which such effects can be forecast.

Standard learned dynamics models conflate these two objects. Fixed-dimensional recurrent and state-space models bind the architecture to one system instance, so a model trained at one subsystem count cannot simply be resized. Graph and message-passing models can process variable layouts, but without explicit storage, dissipation, and power-preserving interconnection they can still produce physically unstable rollouts under target autoregression. Both failures appear in our benchmarks: fixed-size models cannot be reused from a 4-joint to a 6-joint arm without redesign, and in ANYmal-D $\rightarrow$ Go2 transfer a Shared-GRU's rollout error degrades $14\times$ while graph-based baselines diverge.

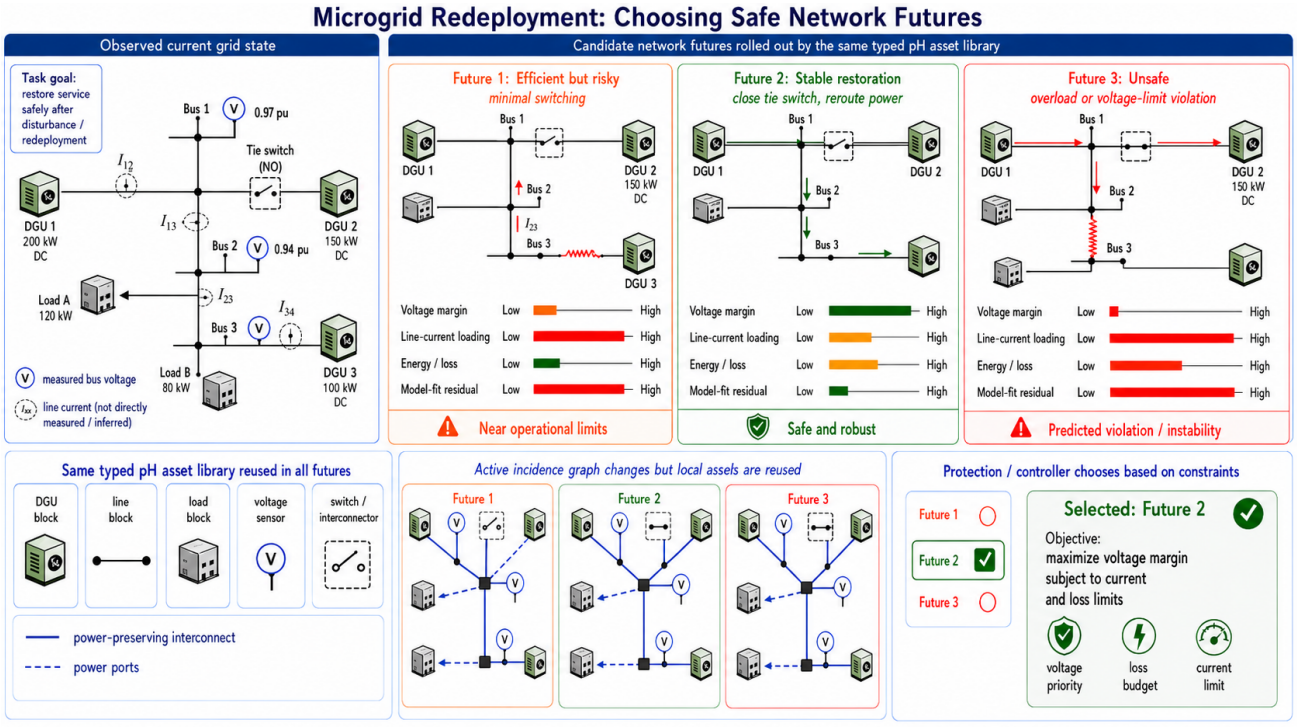

Figure 2. **Microgrid redeployment as candidate-future selection.** The electrical analogue of Figure 1: from the same observed grid state, C-PHAST reuses a typed pH asset library of DGU, line, load, sensor, and switch components to roll out alternative network futures under different active feeder graphs, where the line-to-bus incidence matrix records which assets exchange current. The deployed interface may expose bus voltages while line currents and converter states remain hidden. Each candidate future exposes voltage margin, line-current loading, loss, and model-fit residuals, allowing protection or control logic to select a restoration action that satisfies operational constraints without relearning local asset dynamics. Figure 4 gives the shared typed-block mechanism behind both the robotics and microgrid pictures.

C-PHAST makes the redeployment object explicit. A deployment specification $\mathcal{S}_{\text{target}}$ tells the model which typed subsystems exist, how they are interconnected, what observations are available, how raw coordinates are charted, and which candidate future is being queried. Shared local port-Hamiltonian (pH) blocks encode reusable storage, dissipation, and ports; deployment-specific wiring and explicit inputs determine how blocks exchange power, state, or signals. A rollout returns both predicted coordinates and named physical consequences: energy use, dissipation, port work, support/contact load, residuals, and stability margins.

Here and throughout, *typed* has a concrete physical meaning rather than a programming-language one. A subsystem type is a declared component role such as arm, leg, excitable cell, DGU, or line; a consequence type is a named physical readout such as energy, port work, support load, sensing value, or residual. Blocks with the same type share dimensions and parameters, and readouts with the same type have the same units and downstream interpretation.

The typed boundary also fixes a modeling resolution. A robot can be exposed as joint/link blocks, limb modules, or base, arm, and gripper blocks; changing that choice changes which local Hamiltonians are instantiated and how many rows and columns the interconnection has, while preserving the same open-system port interface. The experiments here instantiate one declared resolution at a time, so the claim is compatibility with multi-resolution composition rather than automatic learning of a coarse-to-fine hierarchy. In short, C-PHAST makes three commitments: learned local pH dynamics are shared by declared type, deployment-specific structure enters through an explicit system specification, and rollout outputs remain named physical consequences until a downstream objective scores them.

A motivating deployment regime. Consider a home-cleaning robot that can wipe an open counter edge with one arm, but must coordinate two or three identical arm modules to reach a cabinet corner or a narrow gap behind a bin. In the harder cases, one arm applies force at the task surface and generates a reaction wrench on the body, while another arm is reassigned to brace or stabilize the platform. The local arm physics do not change; what changes is which contacts carry support and how power is routed across the body. We use this modular robot as a running example: the visible changes are which

physical parts are active, which contacts carry load, and which measurements are available before the robot commits to a future. This is not only a thought experiment. Sleiman et al. (2023) demonstrate a quadrupedal mobile manipulator opening heavy doors and traversing spring-loaded doors by searching over contact states and contact-switching actions, then tracking the resulting whole-body trajectories with MPC/QP control. Their framework solves the planning-and-control side once the relevant robot and contact dynamics are available. The learned-model question is upstream: can a dynamics model be re-instantiated when the active support/manipulation graph, subsystem count, or embodiment metadata changes, rather than relearning a monolithic system for each contact mode?

The same pattern in excitable systems. The FitzHugh–Nagumo benchmark gives a compact biology-motivated version of the same question. FitzHugh–Nagumo dynamics are a canonical reduced model of excitable biological activity with an electrical-circuit realization (FitzHugh, 1961); related excitable models also appear in cardiac and electronic pattern-forming systems (Cebrián-Lacasa et al., 2024). Here the reusable object is an excitable-cell block, while deployment changes the number of cells and the diffusive coupling graph. This isolates repeated typed-block recomposition outside robotics before the paper turns to heterogeneous DGU/line composition in microgrids.

The same pattern in DC microgrids. Islanded DC microgrids make the same redeployment problem concrete. A protection event can open one feeder and close a tie switch; plug-and-play operation can add or remove a distributed generation unit (DGU). The local asset physics do not change: converters store and dissipate energy, lines carry current, and loads draw power. What changes is the deployed composition: the number of active assets, the feeder graph encoded by the line-to-bus incidence matrix, and the sensing interface available at runtime.

This is exactly the setting faced by operational pipelines for plug-and-play voltage control (Tucci et al., 2018; Sadabadi et al., 2018), secondary current-sharing and average-voltage regulation (Cucuzzella et al., 2019), and dynamic-state-estimation-based protection/control (Liu et al., 2018). Those pipelines depend on a model that can turn boundary measurements into state estimates, voltage/current predictions, and calibrated residuals. That dependency becomes fragile when inverters are proprietary, line parameters drift, switch states change, or current sensors are missing. The learned-model question is whether shared typed DGU and line blocks can be re-instantiated on the new feeder graph while still exposing the voltage, current, loss, and residual quantities that protection and control logic consume.

Across these settings, the redeployment variable changes form: robotics changes the active support/contact graph, FitzHugh–Nagumo circuits change the number of excitable cells and rebuild the diffusive chain coupling, and microgrids change the active feeder graph and sensing interface. In all cases, C-PHAST keeps local block semantics fixed while re-instantiating the deployed composition.

Forecasting contract. We study redeployment as a forecasting problem rather than a fixed-instance prediction problem. A model is trained on a source composition and queried on a related target composition whose subsystem count, embodiment, topology, or observation interface may differ:

$$\hat{z}^{\text{target}}_{t+1:t+h} = f_\theta\big(z^{\text{obs}}_{t-L_h+1:t},\ u_{t-L_h:t+h-1};\ \mathcal{S}_{\text{target}}\big),$$

where z^{obs} is the deployment-time observation history, u collects control or exogenous inputs over the history-and-rollout window, and $\mathcal{S}_{\text{target}}$ is the known specification of the target instance. In words, $\mathcal{S}_{\text{target}}$ tells the model what components exist, what type each component has, how components are connected, what measurements are available, how manifold-valued states are represented in coordinates, and what instance metadata is known. Section 3 formalizes this object as subsystem count, typed layout, interconnection topology, observation interface, deployment chart, and optional metadata.

The same contract becomes concrete in different ways across domains. In the cleaning example, the target specification says which arms/base/support parts are active, which part role each one has, which contacts exchange load, and which encoder/contact/sensing channels are visible. In FitzHugh–Nagumo chains, it instantiates repeated excitable-cell blocks and a diffusive coupling graph. In microgrids, it instantiates DGU and line blocks, an active feeder graph encoded by a line-to-bus incidence matrix, and either full-state or voltage-only sensing. In Isaac-Lab cross-robot transfer, the typed layout stays fixed while embodiment metadata such as mass distribution, geometry, and inertia changes.

Ports as interaction boundaries. Intuitively, a port is the named boundary through which a component is pushed, loaded, actuated, or connected to another component. At a robot joint, torque enters through an actuation port and velocity is the conjugate motion; at a contact, wrench and contact velocity define the exchange; in a microgrid, voltage and current define

the electrical exchange; in FitzHugh–Nagumo circuits, source current and diffusive voltage coupling enter through explicit cell interfaces. The paired quantities multiply to power, so accumulating them over a rollout gives port work. This is why a C-PHAST rollout can say not only that a coordinate changed, but which boundary spent energy, carried load, exchanged power, or approached a limit.

Typed consequence queries. The forecast is not only a coordinate prediction. C-PHAST keeps physically named readouts attached to the rollout: joint motion and support load in robots, voltage and current margins in microgrids, and excitable-cell voltage/recovery dynamics in FitzHugh–Nagumo systems. Given candidate future controls, contact choices, or network actions, C-PHAST rolls out each candidate under the same $\mathcal{S}_{\text{target}}$ contract and emits a typed consequence vector $\rho^{(k)}$.

A downstream decision layer then turns those named readouts into planner or protection factors: energy budgets, support margins, collision barriers, sensing-gain terms, voltage limits, current-sharing errors, or residual thresholds. Operationally, this is the action-card question: if candidate future k is chosen, what will it spend, sense, stress, or risk? Figure 7 illustrates the pipeline from raw observables, to typed consequences, to planner-facing factors, to objective-conditioned action cards; Appendix Tables 38 and 39 give the concrete Spot behavior-descriptor and observable-to-factor instantiation. A changed objective alters factor weights, hard envelopes, or priority rules rather than the learned local dynamics. Formally, C-PHAST supports a *constrained selection over controlled flows*. From a common inferred state and deployment specification $\mathcal{S}_{\text{target}}$, each candidate a — a discrete mode (coupling graph, contact/support schedule, or switch configuration) with a control sequence — induces a predicted rollout flow: writing $g_\ell = G_\theta^{(\mathcal{S}_\kappa)}(\cdot, u_\ell)$ for the one-step port-Hamiltonian transition under control u_ℓ, the flow

$$T_a \;=\; g_{H-1} \circ \cdots \circ g_0 \;:\; \hat{x}_0 \mapsto \hat{x}_H(a)$$

composes H one-step maps into a composed transition map—a diffeomorphism in a fixed smooth mode where each substep is a C^1 diffeomorphism on the operating region—with a typed consequence stream $\rho_{\ell+1}(a)$ read out along it. A downstream objective keeps the feasible futures and selects one,

$$a_w^\star \in \operatorname*{arg\,min}_{a \in \mathcal{A}_{\text{feas}}} J_w(a), \qquad J_w(a) = \Psi_G\big(\hat{x}_H(a)\big) + \Delta t \sum_{\ell=0}^{H-1} \sum_q w_q \, \bar{\ell}_q\big(\rho_{\ell+1}(a)\big), \tag{1}$$

where the feasible set $\mathcal{A}_{\text{feas}}$ imposes goal, safety, and resource envelopes on the flow, Ψ_G scores goal attainment, and the nonnegative weights $\{w_q\}$ trade off normalized consequence factors $\bar{\ell}_q$; here $\hat{x}_H(a) = T_a(\hat{x}_0)$, so both J_w and $\mathcal{A}_{\text{feas}}$ depend on the candidate only through its flow T_a. The model *generates* the candidate flows; the objective, not the model, *selects* among them, so the enacted control is a planner-induced choice rather than a policy stored in the learned dynamics. Section 3 details the flow, the envelopes, and the notation (Table 4). The common structure is simple: robots, excitable circuits, and microgrids have different states and ports, but all contain components that store energy, dissipate energy, exchange power, and violate limits. C-PHAST keeps those quantities named during rollout, so decision logic can trade efficiency, safety, sensing, task progress, or protection requirements without changing the dynamics model.

The compositional insight. The architectural bottleneck is that a monolithic model learns local component behavior and system wiring as one inseparable object. C-PHAST separates these roles by learning typed local blocks for what one component does locally: how it stores energy, dissipates energy, receives inputs, and exposes ports. The deployment specification then handles the part that changes across systems, namely how many such blocks exist and which ports are connected. This same separation appears across all of our benchmarks: 1-DoF joint blocks in the planar arm, 3-DoF leg blocks in the legged robots, repeated FitzHugh–Nagumo excitable-cell blocks, and typed DGU/line blocks in microgrids. At the chosen resolution, changing the robot body, cell count, or feeder graph changes the deployed composition, not the meaning of a typed local block. Mechanical benchmarks instantiate this through learned skew-symmetric coupling, FitzHugh–Nagumo circuits through diffusive voltage exchange and source current, and microgrids through line-to-bus incidence wiring.

We introduce **C-PHAST** (Compositional Port-Hamiltonian Architecture for Structured Temporal Dynamics), which turns this separation into a deployable model: shared local port-Hamiltonian blocks, deployment-specific ports and coupling, and an observation-to-state front end that maps available histories into a latent phase-space state before structure-preserving rollout. The concrete source size differs by benchmark, but the transfer contract is common: re-instantiate the learned block library on new subsystem counts or new system instances without retraining, assuming a stable typed decomposition. That decomposition can be one repeated homogeneous block type or a fixed typed-block library with shared within-type

Table 1. **C-PHAST knowledge regimes.** The compositional architecture and rollout pipeline are fixed across all three regimes; only the level of physical specification changes.

Regime	Specified or supplied	Learned from data
KNOWN	local physics, coupling rule, and target layout/topology	nothing, except minor calibration or interfaces
PARTIAL	typed block structure and available target layout/topology	uncertain local physics, damping, coupling corrections, or interfaces
LAW-UNKNOWN	compositional scaffold and declared subsystem layout	local physics and coupling almost entirely

dimensions (formalized in Section 3.7). C-PHAST builds on PHAST (Bhardwaj & Bajaj, 2026), extending its monolithic port-Hamiltonian framework into a compositional transfer architecture. Figures 1 and 2 give the two domain-facing motivations for that contract: task-driven recomposition in robotics and digital-twin redeployment under topology and sensing change in microgrids. Section 3 then makes the shared local port-Hamiltonian and deployed coupling constraints explicit, with Figure 4 abstracting both communities into the same rollout contract.

Specification regimes. The same C-PHAST architecture can be instantiated with different amounts of physical prior knowledge. These regimes describe how much of the model is specified analytically, not which deployment stress is being tested. KNOWN specifies the local laws and coupling rule analytically, then instantiates them on the supplied target layout or topology, up to minor calibration or observation-interface fitting when required. PARTIAL keeps the declared typed block structure and any available target layout or topology, but learns uncertain pieces such as constitutive laws, damping terms, coupling corrections, or the observation-to-state map. LAW-UNKNOWN keeps only the compositional scaffold and learns the local physics and coupling almost entirely from data, while still using the declared subsystem layout needed to instantiate blocks. We deliberately name this regime LAW-UNKNOWN (declared-layout / law-unknown) rather than "unknown": the local constitutive laws and coupling are learned, but the typed decomposition and target layout are still supplied, so it is not a decomposition- or topology-discovery setting. This distinction matters most in the electrical benchmarks, where PARTIAL does not mean topology-unaware. Both KNOWN and PARTIAL can receive the same feeder graph at deployment time; what differs is whether the local DGU/line laws are fixed analytically or partly learned. In the voltage-only microgrid benchmark, for example, PARTIAL keeps the DGU/line typed layout and known incidence pattern while learning the deployable observation-to-state interface. Table 1 summarizes the distinction.

Benchmark roles. The knowledge regimes describe what the model knows or learns; the benchmark roles describe which deployment stress each domain tests. With that distinction in place, the experiments form an evidence map rather than interchangeable demonstrations.

Robotics carries the rollout, decision-facing, and hardware-facing part of the claim. Isaac Lab tests cross-robot pH rollout, variable-N deployment, and matched-state candidate-future scoring under embodiment and contact-related changes. Spot-with-arm logs add the real robot telemetry test: from six teleoperated episodes, C-PHAST maps hardware histories and behavior descriptors to named future consequences that become objective-conditioned action cards. Together, Isaac and Spot show the two sides of the robotics interface: branchable simulation rollouts for comparing candidate futures, and real-world consequence forecasting from physical robot data.

FitzHugh–Nagumo circuits provide the compact non-robot repeated-block check. Because this canonical excitable-systems model has an electrical circuit interpretation and broader use in biological and electronic pattern-forming systems (FitzHugh, 1961; Cebrián-Lacasa et al., 2024), it lets us test a different physical family while keeping the protocol precise: train one shared excitable-cell block at $N{=}3$, rebuild the diffusive coupling graph, and transfer to unseen cell counts.

DC microgrids then move from repeated homogeneous blocks to heterogeneous electrical composition. Their DGU-count transfer is motivated by plug-and-play operation (Tucci et al., 2018; Sadabadi et al., 2018); their feeder-graph reconfiguration and sparse-sensing protocols test deployment changes that matter for digital twins, protection, and monitoring (Liu et al., 2018). The resulting picture is deliberately bounded: N-PHDAE remains the stronger raw forecaster in the full-state known-graph protocol, topology reconfiguration is where topology-aware C-PHAST PARTIAL becomes the tighter design surrogate, and sparse sensing shows a competitive degradation pattern together with typed residual channels for monitoring.

Why port-Hamiltonian structure? Compositional architectures alone (e.g., message-passing networks, shared-weight GRUs) can process variable N, but they do not transfer well: on Isaac Lab, Shared-GRU degrades $14\times$ despite being

compositional. What distinguishes port-Hamiltonian structure is the energy account it attaches to each typed block. The block stores energy in H; it can lose energy through nonnegative dissipation; and it can gain energy only through declared ports where an input and output form a power pair. In mechanics this pair can be torque and velocity; in circuits it can be current and voltage; in contact or support reasoning it can be wrench and boundary motion. The PHASTCore column in Figure 4 is a compact version of this local accounting rule:

$$\frac{dH}{dt} \leq y^\top u,$$

where u and y are conjugate port variables and $y^\top u$ is the power entering through the boundary. The missing part of the inequality is the dissipated power, so an unforced block cannot create stored energy internally. The model therefore predicts a physical balance: what entered through ports, what remained stored, what was dissipated, and what was exchanged internally without creating net energy. This matters for composition because a power-preserving interconnection sends what leaves one block into another with the opposite sign. When the connected blocks are summed, internal exchange cancels in the total energy balance; only external ports, domain inputs, and dissipation remain. Changing the body or topology therefore changes where energy flows, not the local accounting rule. In our experiments, this structural bias is associated with smaller transfer gaps and more stable long-horizon rollouts. We treat that rollout behavior as empirical rather than theorem-level: a controlled four-way comparison suggests that port-Hamiltonian structure reduces the cross-robot transfer gap from $9\times$ to $1.4\times$, while compositionality without pH structure is insufficient (Section 4).

1.1. Contributions

C-PHAST makes four connected contributions.

1. **Redeployable typed pH block library.** C-PHAST learns typed local port-Hamiltonian blocks for reusable storage, dissipation, and port behavior, while the deployment specification supplies subsystem count, typed layout, topology, observations, charts, and metadata. The same typed block library can be re-instantiated across subsystem count, embodiment metadata, topology, and observation-interface changes, yielding one constrained forecasting class rather than one monolithic dynamics model per system instance.

2. **Energy-accounting structure for transfer.** C-PHAST combines pH local blocks with power-preserving interconnection so that stored energy can change only through declared ports, domain inputs, or dissipation in the continuous-time idealization. A controlled four-way comparison separates this structure from compositionality alone: pH structure reduces the cross-robot transfer gap from $9\times$ to $1.4\times$, and on Isaac Lab transfer (ANYmal-D $\rightarrow$ Go2) C-PHAST's 50-step rollout error degrades only $1.5\times$ while unconstrained baselines degrade 9–$14\times$ or diverge.

3. **Typed consequence interface before scoring.** C-PHAST turns learned dynamics into a query interface that exposes named physical consequences before any scalar reward, cost, or veto is formed. Given candidate controls or support/contact hypotheses, it predicts quantities such as energy use, dissipation, port work, support/contact load, residuals, sensing value, and stability margins. This paper demonstrates that interface in finite-candidate scoring, a Go2 lateral-push physical critic, and real Spot-with-arm physical-consequence forecasting from logged possible-next-behavior windows; changing the downstream objective changes factor weights, envelopes, or priority rules rather than the learned dynamics.

4. **Deployment-contract guarantees.** We make the redeployment claim quantitative rather than asserted: a finite-step energy-balance defect (third order in Δt), an observer-aware transfer bound decomposing local, edge, port, and numerical error, and typed-consequence decision margins (ranking, regret, robust feasibility). Their constants are theorem hypotheses—certified for the analytic known-operator deployments and empirically calibrated for trained or hybrid ones—so the guarantees are conditional and explicitly tiered, not universal (Section 3.8).

These contributions rest on assumptions that we foreground here because they bound what "redeployment" and "multiscale" mean in this paper. C-PHAST assumes a finite typed block library with fixed within-type dimension, a declared component decomposition, and transfer that respects that typed decomposition; the mechanical robot-dynamics experiments additionally assume actuated-coordinate-only (q-only) rollout and approximately separable Hamiltonians, and all guarantees are stated for the exact-port continuous-time idealization, leaving a finite-step, observed-coordinate gap. Within these assumptions, the present experiments evaluate repeated-block and finite typed-library transfer under changes in subsystem count, embodiment

metadata, topology, and observation interface, with Spot providing the real-world physical-consequence forecasting and action-card result; C-PHAST does not discover decompositions, topologies, or new component types.

Section 2 reviews related work. Section 3 presents the C-PHAST architecture, including compositional pH dynamics, canonicalization, and the q-only pipeline. Section 4 evaluates transfer across subsystem count, embodiment, topology, and observation interface on mechanical and electrical benchmarks, and foregrounds the real Spot-with-arm consequence-forecasting result. Section 5 briefly discusses control compatibility with the PHAST-style Energy–Casimir template. Section 6 discusses limitations and future directions.

2. Related Work

Related work is organized by what object a method learns and what it exposes at deployment. C-PHAST learns reusable typed port-Hamiltonian (pH) blocks, rebuilds them on a supplied deployment specification, and exposes rollout consequences before objective scoring. The experiments separate the roles of that contract: robotics tests cross-embodiment and decision-facing transfer, FitzHugh–Nagumo circuits isolate repeated excitable-cell recomposition outside robotics, and microgrids test heterogeneous DGU/line composition against the closest known-graph electrical baseline under topology and sensing changes.

Fixed-system world models and physical priors. World models for model-based RL (Ha & Schmidhuber, 2018; Hafner et al., 2023; Hansen et al., 2024) and robotic world models such as RWM (Li et al., 2025) use learned rollouts for planning or policy optimization. Dreamer, TD-MPC2, JEPA-style predictive models, and stable-worldmodel infrastructure all support the standard loop in which a learned predictor is consumed by a planner, value objective, CEM, or MPPI solver (Zhou et al., 2024; Maes et al., 2026). These systems can be strong forecasters, but the predictor is usually tied to one input/output layout, and the planner-facing signal is commonly a latent state, value, reward, or auxiliary diagnostic. C-PHAST keeps the solver-facing role but changes the queried object: each candidate future returns named physical consequence channels and hard-envelope factors before any scalar objective is formed. Fixed-system physics priors have the complementary limitation. Hamiltonian Neural Networks (Greydanus et al., 2019), Lagrangian Neural Networks (Cranmer et al., 2020), energy-prior robot models such as Deep Lagrangian Networks (Lutter et al., 2019), and FeLaN (Viernickel et al., 2025) encode useful physical structure for one embodied system. PHAST (Bhardwaj & Bajaj, 2026) is the direct monolithic predecessor of this work: it provides explicit pH structure and q-only deployment, while C-PHAST changes the reusable object from one fixed system model to a typed local block library.

Robot adaptation, candidates, and contact modes. Robotics has a mature policy-side literature on deployment shift. RMA and A-RMA infer deployment latents from recent proprioceptive history and adapt the policy to estimator mismatch (Kumar et al., 2021; 2022); Walk These Ways exposes human-tunable behavior parameters (Margolis & Agrawal, 2023); shared modular and morphology-agnostic policies reuse actuator-level modules or route observations through embodiment descriptors (Devin et al., 2017; Huang et al., 2020; Bohlinger et al., 2025). These methods expose structure at the policy layer. C-PHAST makes the analogous commitment at the dynamics layer: the deployment specification selects typed local blocks and a coupling pattern for a reusable forward model. Multi-contact loco-manipulation and recent whole-body systems show where such a model is consumed. Sleiman et al. (Sleiman et al., 2023) search over contact schedules and track whole-body trajectories with MPC/QP control; SUMO samples high-level commands around a pretrained controller (Zhang et al., 2026); SafeMimic filters sampled futures with learned safety values (Bahety et al., 2025); and SLAC learns a low-dimensional latent action space for real-world whole-body RL (Hu et al., 2025). These papers address candidate generation, contact planning, control, or safety filtering once candidate dynamics can be evaluated. C-PHAST supplies the upstream learned-model interface: a planner, sampler, demonstrator, or latent-action policy proposes futures, while C-PHAST rolls them out through typed physical channels such as energy, dissipation, support/contact, port work, residuals, and envelope violations.

Compositional dynamics without physical accounting. Message Passing Neural Networks (Gilmer et al., 2017), Graph Network-based Simulators (Sanchez-Gonzalez et al., 2020), MeshGraphNets (Pfaff et al., 2021), and DeepSets (Zaheer et al., 2017) support variable-size inputs by sharing local update rules across nodes or pooled elements. This is the right architectural direction for transfer across subsystem count. The missing ingredient is a physical account of what the shared update may do: storage inequalities, nonnegative dissipation, and power-preserving interconnection laws are not built into the rollout. Rollout robustness is therefore empirical rather than inherited from structure. In our Isaac Lab experiments, this

distinction is concrete: graph/message-passing baselines can process the target robot layout, but diverge under cross-robot rollout, while pH-structured models remain bounded.

Port-Hamiltonian learning and composition. Port-Hamiltonian systems (Van Der Schaft & Jeltsema, 2014) separate conservative exchange, dissipation, and external power ports explicitly. Port-Hamiltonian neural networks (Desai et al., 2021), dissipative Hamiltonian variants (Sosanya & Greydanus, 2022; Zhong et al., 2020), and Lie-group pH neural ODEs (Duong et al., 2024) show that learned dynamics can benefit from storage and damping structure. A parallel line learns pH dynamics from *partial or noisy observations* through an encoder that reconstructs latent state from a past input–output window: OE-pHNN (Moradi et al., 2025) identifies a continuous-time pH model under output-error measurement noise with a subspace encoder, and CIPHER (Li et al., 2026) first learns latent dynamics contrastively from partial (including image) observations and then projects them onto a pH submanifold. Both are *single-system* identification; C-PHAST reuses the same observation-to-state idea in its q-only front end, but attaches it to a typed block library re-instantiated across deployments rather than to one fixed system. For C-PHAST, the pH form is useful for an additional reason: dissipation and external work remain explicit port-level accounting terms, so they can be returned as named consequence channels rather than auxiliary diagnostics. Most learned pH models are still fixed in system dimension. C-PHAST inherits PHAST-style local guarantees, including PSD damping in the pH blocks, while changing the reusable object from a single model to a typed local block family.

The compositional ingredient is standard pH interconnection theory (Van Der Schaft & Jeltsema, 2014; Cervera et al., 2007; Ehrhardt et al., 2026). Composed systems remain port-Hamiltonian when the coupling is power-conserving:

$$J_{\text{coupled}} = \operatorname{diag}(J_1, \ldots, J_s) - \hat{\mathbf{B}}\,\hat{\mathbf{C}}\,\hat{\mathbf{B}}^\top, \quad \hat{\mathbf{C}} = -\hat{\mathbf{C}}^\top. \tag{2}$$

Skew-symmetry ensures zero net coupling power, which standard learned couplings do not preserve automatically. The closest architectural predecessor is the compositional PHNN framework of Neary & Topcu (2023), which learns subsystem pH models independently, composes them through known or learned physics-informed interconnections, and gives a composite modeling-error bound. C-PHAST does not claim that pH subsystems being composable is new. Its added object is the deployment contract around that composition: shared typed blocks are rebuilt under a supplied target specification that can change subsystem count, topology, coordinate chart, observation interface, or embodiment metadata, and the rollout exposes named consequences and residuals for downstream decision layers. Otterdijk et al. (van Otterdijk et al., 2024) study learning subsystem dynamics via pH neural networks when a subsystem is observed as part of a larger coupled system. The closest compositional physical baseline in our electrical setting is N-PHDAE (Neary et al., 2024), which composes per-subsystem PHDAE models through known graph constraints. That composition is realized as algebraic differential-algebraic-equation (DAE) constraints on a fully specified graph, which is well matched to exact electrical conservation laws; C-PHAST instead composes through the power-preserving skew coupling in (2), which keeps the interconnected model an explicit ODE, so a topology or sensing change is absorbed by rebuilding the coupling rather than reconstructing an algebraic constraint system. We therefore treat it as the strongest known-graph electrical baseline rather than as a generic graph-learning comparison: in our microgrid experiments, N-PHDAE remains stronger for full-state known-graph raw rollout. C-PHAST targets a different redeployment contract. It learns a typed pH block library whose instances can be rebuilt when subsystem count, topology, embodiment metadata, coordinate chart, or observation interface changes, and it keeps rollout outputs as typed physical consequences for monitoring, design metrics, and action-card scoring. In the microgrid results, this distinction appears under topology reconfiguration and voltage-only sensing: C-PHAST PARTIAL is a tighter design surrogate under reconfiguration, remains competitive under partial sensing, and exposes residual channels indicating which physical part of the deployment changed.

Microgrid control and protection pipelines. The DC-microgrid literature also clarifies why a learned dynamics surrogate is useful in the first place. Tucci et al. (Tucci et al., 2018) design line-independent plug-and-play primary voltage controllers from local DGU models; Sadabadi et al. (Sadabadi et al., 2018) make that plug-and-play design robust to polytopic local uncertainty and constant-power-load effects; Cucuzzella et al. (Cucuzzella et al., 2019) add secondary current sharing and weighted-average voltage regulation through consensus and sliding-mode control. Liu et al. (Liu et al., 2018) address a different layer: dynamic-state-estimation-based protection and converter feedback, where a high-fidelity protection-zone model produces both a state estimate $\hat{x}$ for control and a χ^2 residual statistic for fault decisions. In these pipelines, the dynamics model is the operational object that makes hidden-state reconstruction, residual calibration, and downstream control/protection decisions possible. These works solve controller or relay logic once the relevant DGU, line, protection-zone, and sensing models are available. C-PHAST targets the upstream learned-surrogate problem: learning typed

Table 2. **Comparison of dynamics learning approaches.** C-PHAST combines variable layout support, energy/passivity structure, PSD damping in the local pH blocks, and compositional interconnection theory. *HNN/LNN conserve energy (lossless) but cannot model dissipation. †D-HNN does not enforce $D \succeq 0$. ‡FeLaN performs inverse dynamics (system ID), not forward prediction. Neary–Topcu PHNN is the closest learned compositional pH predecessor; N-PHDAE targets electrical (DAE) systems and is the closest known-graph electrical baseline. "n/a" indicates not applicable.

Method	Variable Layout	Energy Structure	$D \succeq 0$ by Constr.	Compos. Theory
Dreamer / TD-MPC2 / RWM	✗	✗	n/a	✗
MPNN / GNS / MeshGraphNets	✓	✗	n/a	✗
HNN / LNN	✗	✓*	n/a	✗
D-HNN	✗	✗†	✗	✗
Dissip. SymODEN	✗	✓	✓	✗
PHAST	✗	✓	✓	✗
FeLaN‡	✗	✓	✓	✗
Duong (Lie group pH)	✗	✓	✓	✗
Neary–Topcu PHNN	✓	✓	✓	✓
N-PHDAE	✓	✓	n/a	✓
C-PHAST (ours)	✓	✓	✓	✓

port-Hamiltonian DGU and line blocks that can be re-instantiated when subsystem count, incidence, or observation interface changes. The contribution is the compositional dynamics interface those downstream pipelines require.

Port-Hamiltonian structure on the control side. A related line of work uses port-Hamiltonian structure to parameterize *distributed controllers* rather than plants. Furieri et al. (Furieri et al., 2022) embed compositional pH structure into distributed neural network control policies and prove closed-loop stability irrespective of the interconnection topology and the chosen NN weights; Zakwan and Ferrari-Trecate (Zakwan & Ferrari-Trecate, 2024) extend the result to a finite $\mathcal{L}_2$ gain. These methods assume the plant is a known pH network and learn the controller. The roles are compatible: C-PHAST learns the *plant* while leaving the control layer unchanged, and distributed pH controller parameterizations could consume the learned plant model in a single stack.

Other structure-preserving approaches. SympNets (Jin et al., 2020) and Volume-Preserving Transformers (Brantner et al., 2024) enforce symplectic or volume-preserving structure, which is well matched to conservative systems but not to generic dissipative flows with phase-space contraction. PINNs (Raissi et al., 2019) enforce physics through soft losses; C-PHAST uses architectural constraints for the local pH blocks and for the interconnection law.

Summary. Table 2 summarizes this landscape. Monolithic models can be physically structured or highly expressive, but are tied to one system instance. Compositional models can handle varying system size, but often lack a physical interconnection law. C-PHAST is best read as an extension of prior compositional PHNNs into a deployment-conditioned, partially observed, decision-facing world-model contract, evaluated against known-graph compositional baselines such as N-PHDAE where those baselines are strongest. The table focuses on model structure; candidate-generation and safe-planning methods are consumers of the typed rollout vector.

Table 2 compares model *structure*; it does not show *where* each method is useful. The key observation is that real deployments rarely withhold everything at once: a redeployment typically changes only specific parts of the specification—how many typed blocks are present, how they are wired, or what can be sensed (here *q-only* sensing means positions and commands are available but clean velocities or momenta are not). Figure 3 orders deployment by which part changes, and makes the central point of this paper's positioning visible: these changes *concentrate* in a few columns—subsystem count, topology, and sensing—which is exactly where full-state specialized baselines begin to degrade and C-PHAST's deployment contract becomes the differentiator. Table 3 names the concrete trigger behind each column and the method competitive there, together with the two seams we do not claim: LAYOUT/COUNT carries heavy demand but is contested, and DECOMP. UNKNOWN is open for every method.

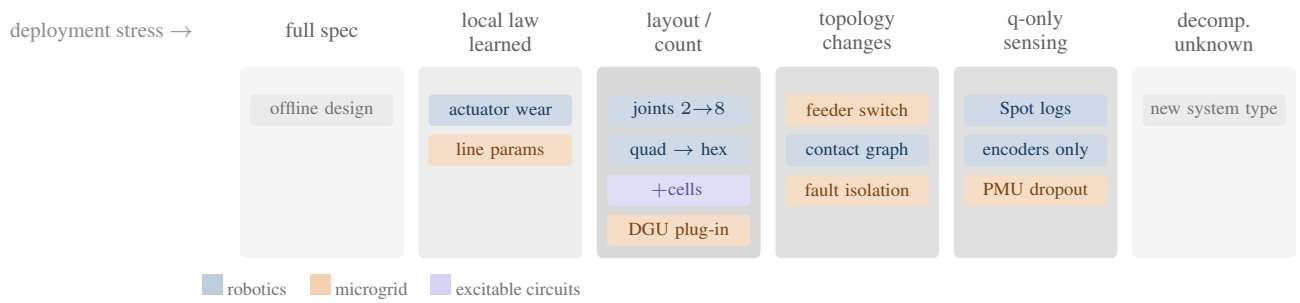

Figure 3. **Real redeployments concentrate.** Ordering deployment by which part of the specification changes, the changes that actually occur across our three benchmark domains pile up in LAYOUT/COUNT, TOPOLOGY CHANGES, and Q-ONLY SENSING (darker columns) rather than spreading uniformly. Table 3 names the method competitive in each column and marks the two honest seams; Appendix K gives the per-cell specification fingerprint.

Table 3. **Where real deployments land.** Concrete redeployment triggers by domain for each deployment-stress column of Figure 3, how often deployments hit it, and the method competitive there. Demand concentrates in LAYOUT/COUNT, TOPOLOGY CHANGES, and Q-ONLY SENSING. Two seams are marked honestly: LAYOUT/COUNT is *contested* (Neary–Topcu is also strong there), so the strongest-demonstrated zone begins at TOPOLOGY CHANGES; and DECOMP. UNKNOWN is open for every method. The exact per-cell specification fingerprint is in Table 50.

Stress column	Demand	Real deployment triggers (domain)	Competitive
full spec	low	offline design / benchmark condition	N-PHDAE; monolithic PHAST
local law learned	low	actuator wear (robot); line parameters (grid)	monolithic PHAST
layout / count	**high**	joints 2→8, legs biped/quad/hex (robot); +cells (excitable); +DGU plug-in (grid)	Neary–Topcu; C-PHAST *(contested)*
topology changes	**high**	feeder switch, fault isolation (grid); contact graph (robot)	**C-PHAST**
q-only sensing	**high**	Spot logs, encoders-only (robot); PMU dropout (grid)	**C-PHAST**
decomp. unknown	low	new system type	*open to all*

3. Method: C-PHAST Architecture

C-PHAST is a deployment-time world-model contract for systems assembled from reusable physical parts. It combines four ingredients: (1) a deployment specification that declares the instantiated parts, observations, and interconnection, (2) shared port-Hamiltonian local dynamics, (3) passivity-preserving skew-symmetric coupling, and (4) an observation front end that maps available measurements into the latent pH state used for rollout. We first state the constrained forecasting problem, then define the compositional port-Hamiltonian model class, and finally show the deployed rollout pipeline.

High-level picture. Use the modular robot from Section 1 as the running deployment scene. One candidate wipes with one arm, another uses a second arm to brace the base, and a third reaches with insufficient support. The local arm/base physics are the reusable part. The changing part is visible in the scene: how many physical parts are active, what role each part plays, who pushes or braces against whom, what the robot can measure, and what instance metadata is known. C-PHAST records those choices in the deployment specification, while shared port-Hamiltonian dynamics advance the typed local blocks. A *type* is a declared physical role with fixed local coordinates, ports, and shared parameters, such as a robot leg block, an excitable-cell block, a DGU block, or a line block. From a short measurement history, after any declared coordinate normalization or angle wrapping, an observer and optional canonicalizer infer a latent pH state for each block. A skew-symmetric coupling layer then exchanges power between blocks without injecting it. Table 6 records how each benchmark domain instantiates this same contract.

From deployment failures to algorithmic components. The failures motivating Sections 1–2 map directly to the pieces of the deployed predictor. Fixed-size monolithic models fail under subsystem-count or topology change, so C-PHAST represents the target instance by a deployment specification $\mathcal{S}_{\text{target}}$. In the running example, this means the same learned arm/base library is queried after the active parts, bracing contacts, or measured channels change. Raw histories are not automatically pH states, and some measured coordinates have units, periodicity, positivity, or normalization conventions that should be declared rather than learned. The deployment chart is this declared coordinate map; together with the observer, it turns available measurements into the latent state used by the dynamics. Unstructured local dynamics can

inject spurious energy under rollout, so each typed local block is a PHASTCore port-Hamiltonian module with nonnegative damping. Generic graph couplings can create or destroy energy, so inter-block exchange is restricted to a skew-symmetric power-preserving operator. The learned object itself remains a rollout model: later sections use the same operator both for open-loop forecasting and for candidate-future scoring.

From PHAST to C-PHAST. PHAST (Bhardwaj & Bajaj, 2026) is the single-system version of this construction: it maps q-only measurement history, after any declared coordinate map, into canonical phase space, advances a port-Hamiltonian dynamics block with learned V, M, and $D \succeq 0$, and reads out the next observation. C-PHAST changes the reusable object. Instead of learning one monolithic PHAST model for one robot or circuit, it learns typed PHASTCore blocks and a deployment-time interconnection rule. Algorithm 1 is the latent compositional PHAST update; Algorithm 2 wraps that update with the q-only observer/canonicalizer interface used in the robotics benchmarks.

Notation. The notation below is the compact version of the deployment scene above. The number of active parts, cells, or grid assets is s; their roles are the type labels α_i; who exchanges load, voltage, or current with whom is G_{target}; what can be measured is $\mathcal{O}_{\text{target}}$; what coordinate normalization or wrapping is applied is χ_{target}; and instance-specific body or asset information is Ξ_{target}. We use x_t for the latent pH state advanced by the model. In mechanical settings $x_t = (q_t, p_t)$, where q_t are generalized coordinates, $p_t \in \mathbb{R}^N$ are latent momenta, and robotic torques are written τ_t when needed. Raw observed coordinates carry a tilde, as in $\tilde{q}_t$. After the optional coordinate map, the model-facing coordinate is denoted by z_t; in mechanical local blocks we then write $q_t = z_t$. In non-mechanical settings, q_t denotes the domain coordinate vector decoded from the pH state, such as excitable-cell variables or electrical state/readout coordinates. Generic external inputs are denoted by u_t. For mechanical q-only settings the observation is just the position history, while $y^{\text{port}} = \dot{q}$ denotes the physical port output, power-conjugate to the input port. The history length is denoted L_h throughout. For any sequence a, the shorthand $a_{r:s}$ denotes the inclusive window $(a_r, \dots, a_s)$. All q-only history equations below use times $t \geq L_h - 1$. Candidate futures are indexed by $k = 1, \dots, K$. For compositional models, s is the number of instantiated subsystems. In homogeneous repeated-block settings each subsystem has the same local DoF d, so $N = s\,d$. In heterogeneous typed-block settings each subsystem has a type α_i in a finite library $\mathcal{T}$ (e.g., $\mathcal{T} = \{\text{DGU}, \text{line}\}$ in the microgrid) with local DoF d_{α_i}, so $N = \sum_{i=1}^{s} d_{\alpha_i}$. We use *typed layout* to mean the assignment $\{\alpha_i\}_{i=1}^{s}$ of block types to the instantiated subsystems, distinct from the interconnection topology. We write

$$\mathcal{S}_{\text{target}} = (s, \{\alpha_i\}_{i=1}^{s}, G_{\text{target}}, \mathcal{O}_{\text{target}}, \chi_{\text{target}}, \Xi_{\text{target}})$$

for the deployment-time specification of a target system instance, comprising subsystem count, typed layout, interconnection topology, observation interface, deployment chart, and optional instance metadata. Within this contract, the typed layout selects which local blocks are instantiated, G_{target} determines how those instantiated blocks are coupled, $\mathcal{O}_{\text{target}}$ determines the measurement channels available to the observer/canonicalizer front end, χ_{target} maps raw observed coordinates into model-facing coordinates, and Ξ_{target} carries any instance-level metadata that is not itself a graph or measurement object (for example robot body parameters in fixed-layout cross-embodiment transfer). For a fixed deployment specification, we write $\hat{\mathbf{C}}^{(\mathcal{S}_{\text{target}})}$ for the instantiated skew-symmetric coupling operator. Depending on the benchmark regime, this operator is induced by learned coupling parameters θ_C, specified by the supplied graph or incidence structure, or formed by combining both. In the homogeneous count-transfer regime this specializes to the size-indexed matrix $\hat{\mathbf{C}}^{(s)}$.

State pipeline. Throughout the method, the observation-to-rollout path is

$$\tilde{q}_{t-L_h+1:t} \xrightarrow{\chi_{\text{target}}} z_{t-L_h+1:t} \xrightarrow{\Phi_{\phi,\psi}} \hat{x}_t \xrightarrow{G_\theta^{(\mathcal{S}_{\text{target}})}} \hat{x}_{t+1} \xrightarrow{\Pi_q, \chi_{\text{target}}^{-1}} \hat{q}_{t+1}^{\text{obs}}.$$

Here $\tilde{q}_{t-L_h+1:t}$ is the raw measurement window and $z_{t-L_h+1:t}$ is the same window after the optional deployment chart has put it in model-facing coordinates. If the chart is the identity, $z_\tau = \tilde{q}_\tau$ for every τ in the window. In q-only mechanical settings we write the latest model-facing coordinate as q_t once it enters the local pH block; the corresponding pH state has coordinates $x_t = (q_t, p_t)$, and prediction uses the inferred state $\hat{x}_t$. The observer/canonicalizer $\Phi_{\phi,\psi}$ fills in the missing velocity or momentum variables from the whole window, the deployed transition advances the inferred latent state, and the readout projects back to model-facing coordinates before decoding to observation space. The decoded observation-space prediction is written $\hat{q}^{\text{obs}}$ when the chart/readout distinction matters; losses and tables abbreviate it as $\hat{q}$ when no ambiguity is possible. This is the naming convention used by the rollout equations and Algorithm 2.

Deployment charts and state refresh. Different domains do not provide coordinates in the same form. A robot joint angle wraps around a circle, a bus voltage is often reported in per-unit scale, and a quantity constrained to be positive should not be treated as an unconstrained Euclidean number. The chart χ_{target} is the lightweight map in the deployment specification that converts these measured coordinates into the local coordinates used by the typed pH blocks. For a q-only history window and each $\tau = t - L_h + 1, \dots, t$,

$$z_\tau = \chi_{\text{target}}(\tilde{q}_\tau), \qquad \tilde{q}_\tau = \chi_{\text{target}}^{-1}(z_\tau), \tag{3}$$

where χ_{target} can factor over declared block or port coordinates. The identity chart is used when the raw coordinates are already Euclidean. Non-identity charts handle wrapped angles, positive-state transforms, unit normalization, and per-unit electrical scaling. The chart acts only on coordinates: graph and topology information stays in G_{target}, available measurements stay in $\mathcal{O}_{\text{target}}$, and instance metadata stays in Ξ_{target}. Inside the pH transition we write q for the model-coordinate component. Predictions are decoded through $\chi_{\text{target}}^{-1}$ before observation-space errors are evaluated. At deployment time, a new measurement window can refresh the latent state by reapplying the same chart, observer, and canonicalizer with fixed learned weights. This is state refresh, not adaptive control or online parameter learning. A fully measurement-closed rollout, where new observations repeatedly re-anchor the prediction during execution, is left as follow-up work.

3.1. Problem Statement

We formalize the dynamics learning problem addressed by C-PHAST. The compatible Energy–Casimir control extension is discussed separately in Section 5.

Constrained forecasting problem. In the running robot example, training on one composition and querying another means that the learned arm/base block library is fixed, but the active parts, bracing graph, available measurements, or body metadata may change. The same pattern becomes cell count and diffusive links in FitzHugh–Nagumo circuits, and DGU/line layout, feeder incidence, and sensing level in microgrids. Let

$$\mathcal{D} = \big\{(\tilde{q}^{(i)}_{0:T_i}, u^{(i)}_{0:T_i-1}, \mathcal{S}^{(i)})\big\}_{i=1}^{|\mathcal{D}|}$$

be observed coordinate trajectories from compositional systems, each paired with its deployment specification $\mathcal{S}^{(i)}$. For robotics, $\tilde{q}_t$ is the q-only generalized-coordinate history. For FitzHugh–Nagumo circuits and microgrids, $\tilde{q}_t$ denotes the domain coordinate or observed readout vector selected by the benchmark observation interface. In homogeneous settings those subsystems share a common local dimension d, so $N_i = s_i d$. In heterogeneous typed-block settings they are drawn from a finite type library $\mathcal{T}$ with type-dependent local dimensions, so $N_i = \sum_{j=1}^{s_i} d_{\alpha_j}$. Here $\tilde{q}_t \in \mathbb{R}^{N_i}$ denotes noisy or partial observed coordinates/readouts, and u_t collects external actuation or exogenous inputs when present (with $u_t \equiv 0$ in autonomous benchmarks). We learn shared parameters

$$\theta = (\theta_{\text{loc}}, \theta_C, \phi, \psi), \qquad \theta_{\text{loc}} = \big\{(\theta_V^{(\alpha)}, \theta_M^{(\alpha)}, \theta_D^{(\alpha)})\big\}_{\alpha \in \mathcal{T}},$$

where θ_{loc} parameterizes the shared local Hamiltonian and damping modules, one set per block type $\alpha \in \mathcal{T}$ (with $|\mathcal{T}| = 1$ in homogeneous settings), θ_C parameterizes learned or partially specified skew-symmetric coupling templates, ϕ parameterizes the observer, and ψ parameterizes the canonicalizer when one is needed. When a benchmark specifies the interconnection analytically, the corresponding coupling entries are held fixed or omitted from optimization; we keep θ_C in the tuple for uniform notation. For each valid training window (i, t), define the operator chain

$$\begin{aligned}
z_\tau^{(i)} &= \chi_{\mathcal{S}^{(i)}}\big(\tilde{q}_\tau^{(i)}\big), \\
\hat{x}_t^{(i)} &= \Phi_{\phi,\psi}\big(z^{(i)}_{t-L_h+1:t}, u^{(i)}_{t-L_h:t-1}\big), \\
\hat{x}_{t+1}^{(i)} &= G_\theta^{(\mathcal{S}^{(i)})}\big(\hat{x}_t^{(i)}, u_t^{(i)}\big), \\
\hat{q}_{t+1}^{\text{obs},(i)} &= \chi_{\mathcal{S}^{(i)}}^{-1}\big(\Pi_q(\hat{x}_{t+1}^{(i)})\big).
\end{aligned}$$

where $\Phi_{\phi,\psi}$ is the history-to-state front end, $G_\theta^{(\mathcal{S}^{(i)})}$ is the deployed one-step latent transition induced by the shared local pH blocks and the coupling instantiated from $\mathcal{S}^{(i)}$, and Π_q extracts the model-coordinate readout from the latent state before decoding through the deployment chart. The next subsections build the deployed transition operator $G_\theta^{(\mathcal{S})}$ in stages; the q-only front end $\Phi_{\phi,\psi}$ is returned to explicitly in Section 3.6. Writing

$$\mathcal{I} := \big\{(i, t) : L_h - 1 \le t < T_i\big\}$$

for the set of valid one-step training windows, the base empirical objective is

$$\min_{\theta_{\mathrm{loc}},\theta_C,\phi,\psi} \ \frac{1}{|\mathcal{I}|}\sum_{(i,t)\in\mathcal{I}} \big\|\tilde{q}_{t+1}^{(i)} - \hat{q}_{t+1}^{\mathrm{obs},(i)}\big\|^2, \tag{4}$$

This is the base one-step supervised term used throughout the paper, with the norm evaluated on the supervised channels declared by the benchmark observation interface. When hidden states are available for analysis but not for deployment, we report them as additional diagnostics rather than as extra decision-time inputs. Below we augment the base term with optional energy, observer, passivity, and rollout regularizers when such supervision is available, subject to three structural constraints:

1. **Port-Hamiltonian local blocks.** Each repeated subsystem belongs to a port-Hamiltonian model class with an energy function, nonnegative dissipation, and explicit input/output ports.
2. **Power-preserving interconnection.** Inter-subsystem coupling is skew-symmetric, so the ideal continuous-time interconnection injects zero net coupling power.
3. **Cross-instance generalization.** The *same* $(\theta_{\mathrm{loc}}, \theta_C, \phi, \psi)$ produces predictions for new deployment specifications $\mathcal{S}_{\text{target}}$ without retraining: in homogeneous settings by replicating one shared local block and instantiating a size-s' coupling matrix $\hat{\mathbf{C}}^{(s')}$, in fixed-layout cross-embodiment settings by holding the typed layout fixed while changing Ξ_{target}, and in heterogeneous settings by reusing the same per-type local modules on a new typed layout or topology.

Constraints (1)–(2) are enforced architecturally (not as soft losses). Constraint (3) distinguishes C-PHAST from fixed-dimension models, whose input/output projections are tied to one source system instance and must be rebuilt or retrained for a new subsystem layout.

Decision problem: selecting a controlled flow. Equation (4) learns the transition operator; at deployment the *same* operator generates a controlled flow for each candidate, which a downstream objective selects among — the objective lives downstream of the model, not inside it. A candidate future $a = (\kappa, u_{0:H-1})$ pairs a discrete mode κ — a deployed coupling graph G_κ, contact/support schedule, or feeder-switch configuration that instantiates $\mathcal{S}_\kappa$ and its operator $G_\theta^{(\mathcal{S}_\kappa)}$ — with a continuous control sequence $u_{0:H-1}$ over a horizon H. From a common inferred state $\hat{x}_0$, its rollout is a composed flow of the port-Hamiltonian system: with $g_\ell = G_\theta^{(\mathcal{S}_\kappa)}(\cdot, u_\ell)$ the one-step transition under control u_ℓ,

$$\hat{x}_{\ell+1}(a) = g_\ell\big(\hat{x}_\ell(a)\big), \qquad T_a = g_{H-1}\circ\cdots\circ g_0 \ : \ \hat{x}_0 \mapsto \hat{x}_H(a),$$

so, for a fixed smooth mode in which each numerical substep is a C^1 diffeomorphism on the operating region, T_a composes H diffeomorphisms and is itself one; where a substep can be singular—strong explicit damping, contact or reset maps, mode changes, wrapped charts, or nonsmooth learned modules—T_a is instead a composed transition map. (Dissipation makes the continuous flow contract the shaped storage, not phase-space volume: the divergence of $(J-R)\nabla H$ depends on $\nabla^2 H$ and on $\nabla J, \nabla R$, and is not sign-definite from $R \succeq 0$ alone.) Along the flow, C-PHAST emits the typed consequence vector $\rho_{\ell+1}(a) = \mathcal{O}_{\text{target}}\big(\hat{x}_{\ell+1}(a), u_\ell; \mathcal{S}_\kappa\big)$ — energy use, dissipation, port work, support/contact load, residual, sensing value, and stability margin — *before* any scalar score is formed. A downstream planner or protection layer then turns these named readouts into a feasibility test and a score.

Admissible flows (hard envelopes). The feasible set

$$\mathcal{A}_{\text{feas}} = \Big\{ a : \ \hat{x}_H(a) \in \mathcal{X}_G,\ u_\ell \in \mathcal{U},\ c_j\big(\rho_{\ell+1}(a)\big) \le 0 \ \forall j,\ell,\ P_\ell(a) \le P_{\max},\ \textstyle\sum_\ell P_\ell(a)\,\Delta t \le E_{\max},\ \sum_\ell r_\ell(a)\,\Delta t \le R_{\max} \Big\}$$

keeps only candidates whose flow reaches a goal set $\mathcal{X}_G$ within the horizon while respecting admissible controls $\mathcal{U}$, pointwise safety constraints $c_j \le 0$, a peak-power cap $P_{\max}$, a total-energy budget $E_{\max}$, and a cumulative-risk cap $R_{\max}$.

Selecting a flow (soft factors). Among admissible flows, the downstream objective of Eq. (1) — a terminal cost Ψ_G scoring goal attainment plus a weighted sum of normalized consequence factors $\bar{\ell}_q\big(\rho_{\ell+1}(a)\big)$ with nonnegative weights $\{w_q\}$ (progress, energy-per-progress, dissipation, smoothness, sensing value, soft risk) — selects one. Table 4 collects the notation.

The problem is hybrid and nonconvex, so $a_w^\star$ is a *local* optimum — a locally optimal admissible flow; the enacted control is receding-horizon, $\pi_w = u_0(a_w^\star)$, a planner-induced feedback law rather than a policy stored in G_θ. Figures 10 and 11

Symbol	Meaning
$a = (\kappa, u_{0:H-1})$	Candidate future: discrete mode κ with continuous control sequence $u_{0:H-1}$ over horizon H.
κ	Deployed mode — coupling graph, contact/support schedule, or feeder-switch configuration — fixing $\mathcal{S}_\kappa$ and $G_\theta^{(\mathcal{S}_\kappa)}$.
$\hat{x}_\ell(a)$	Predicted latent state at step ℓ under candidate a; Δt is the step increment.
$T_a = g_{H-1} \circ \cdots \circ g_0$	Controlled rollout flow (composed transition map; a diffeomorphism in a fixed smooth mode with C^1 substeps); composition of one-step maps $g_\ell = G_\theta^{(\mathcal{S}_\kappa)}(\cdot, u_\ell)$.
$\rho_{\ell+1}(a)$	Typed consequence vector from $\mathcal{O}_{\text{target}}$ (energy use, dissipation, port work, support/contact load, residual, sensing value, stability margin).
$\mathcal{A}_{\text{feas}}$	Feasible set: candidates whose flow satisfies every hard envelope.
$\mathcal{X}_G$, $\mathcal{U}$	Goal set; set of admissible controls.
c_j	Pointwise safety constraints ($c_j \le 0$ at every step).
$P_{\max}, E_{\max}, R_{\max}$	Peak-power, total-energy, and cumulative-risk budgets (P_ℓ power, r_ℓ risk rate).
J_w, Ψ_G	Objective (weighted normalized factors) and terminal cost (goal attainment).
w_q, $\bar{\ell}_q$	Nonnegative factor weights; normalized soft factors.
$a_w^\star$	Selected candidate: a *local* minimizer of J_w over $\mathcal{A}_{\text{feas}}$.
$\pi_w = u_0(a_w^\star)$	Receding-horizon feedback law (first control of the selected candidate).

Table 4. Notation for the decision problem of Eq. (1).

display a finite candidate set $\{a^{(k)}\}$: the same inferred state with different declared interventions, ranked by Eq. (1) among the feasible ones (globally over the drawn candidates, not over the continuum). Because only the weights $\{w_q\}$, envelopes c_j, and budgets change between objectives, never the learned dynamics, a changed objective re-scores the *same* flows. A complementary energy-shaping realization (Section 5) instead *generates* a controlled flow directly, as a storage-shaping feedback law: goal and risk become the minimum and injected damping of a shaped storage on the Casimir manifold, and the closed loop is a dissipative port-Hamiltonian flow for which the shaped storage is a Lyapunov function. It is local for the same reason — convergence to a *local* minimum of the shaped storage; global convergence would require a global argument (properness, forward completeness, a unique relevant critical point, and detectability of the zero-dissipation set), which safety barriers and nonconvex landscapes preclude in general.

Method roadmap. From here, the method has one spine. We first define a single port-Hamiltonian block. We then connect many blocks with a skew-symmetric interconnection, so power exchanged inside the composition cancels. Finally, we wrap the composed model with the deployed pipeline that maps observations into model-coordinate histories and latent states, rolls them forward, and decodes predictions back to observed coordinates. The main text focuses on this invariant predictor contract; the appendix gives the benchmark-specific details for planar-arm count transfer, legged robot transfer, FitzHugh–Nagumo cell-count transfer, and microgrid DGU/line topology and sparse-sensing tests.

3.2. Port-Hamiltonian Dynamics

A port-Hamiltonian (pH) system (Van Der Schaft & Jeltsema, 2014) with state $x \in \mathbb{R}^n$, Hamiltonian $H(x)$ (total energy), skew-symmetric structure $J = -J^\top$, dissipation $R = R^\top \succeq 0$, and port matrix $\mathbf{B}$ evolves as:

$$\dot{x} = (J - R)\nabla H(x) + \mathbf{B}u, \quad y^{\text{port}} = \mathbf{B}^\top \nabla H(x). \tag{5}$$

The key property is passivity: $\frac{dH}{dt} = -\nabla H^\top R \nabla H + (y^{\text{port}})^\top u \le (y^{\text{port}})^\top u$. Energy can only decrease through dissipation or increase through external ports, which is the foundation for compositional stability. For mechanical subsystems with state $x = (q, p)$, torque input $u = \tau$, and $\mathbf{B} = [0; I]$, the port output is $y^{\text{port}} = \partial H/\partial p = \dot{q}$ (velocity).

3.3. Compositional Coupling via Skew-Symmetric Interconnection

In the running robot example, adding a bracing arm should move load through the body, not create energy from the wiring itself. The same principle applies when an FHN cell exchanges voltage with a neighbor or when a microgrid line carries current between buses. Following the port-Hamiltonian interconnection framework (Ehrhardt et al., 2026), we first expose a *coupling port* on each local subsystem and defer other benchmark-specific ports to Eq. (12). For subsystem i, write

$$\dot{x}_i = (J_i - R_i)\nabla H_i(x_i) + \mathbf{B}_i \hat{u}_i, \qquad \hat{y}_i = \mathbf{B}_i^\top \nabla H_i(x_i), \tag{6}$$

where $\hat{u}_i$ and $\hat{y}_i$ are the input and output of the inter-subsystem coupling port. When a subsystem also has actuation, load, contact, or environment ports, we write the total port input as

$$u_i = u_i^{\text{int}} + u_i^{\text{ext}}, \qquad u_i^{\text{int}} := \hat{u}_i, \tag{7}$$

where only u_i^{int} is constrained by the skew interconnect. The internal interconnect is power-preserving; external ports are exactly the channels through which controls or the environment can inject or remove energy. Stack the subsystem states, Hamiltonians, and port variables as

$$x = [x_1; \dots; x_s], \qquad H_{\text{total}}(x) = \sum_{i=1}^{s} H_i(x_i), \qquad \hat{u} = [\hat{u}_1; \dots; \hat{u}_s], \qquad \hat{y} = [\hat{y}_1; \dots; \hat{y}_s],$$

and let $\hat{\mathbf{B}} := \operatorname{diag}(\mathbf{B}_1, \dots, \mathbf{B}_s)$, $\hat{J} := \operatorname{diag}(J_1, \dots, J_s)$, and $\hat{R} := \operatorname{diag}(R_1, \dots, R_s)$. The interconnection is then imposed by the skew-symmetric coupling law; at this stage we retain only the internal coupling ports and temporarily set aside the external inputs u_i^{ext}, which are reintroduced in Eq. (12).

$$\hat{u} = -\hat{\mathbf{C}}^{(s)}\, \hat{y}, \qquad \hat{\mathbf{C}}^{(s)} = -(\hat{\mathbf{C}}^{(s)})^{\top}. \tag{8}$$

Substituting $\hat{y} = \hat{\mathbf{B}}^{\top} \nabla H_{\text{total}}$ into the stacked subsystem dynamics gives the composed closed-port system

$$\dot{x} = \left(\hat{J} - \hat{\mathbf{B}}\hat{\mathbf{C}}^{(s)}\hat{\mathbf{B}}^{\top} - \hat{R}\right) \nabla H_{\text{total}}(x). \tag{9}$$

The three terms in Eq. (9) have distinct roles: $\hat{J}$ is the block-diagonal local pH structure, $\hat{\mathbf{B}}\hat{\mathbf{C}}^{(s)}\hat{\mathbf{B}}^{\top}$ is the inter-subsystem exchange induced by the coupling ports, and $\hat{R}$ collects local dissipation. Because $\hat{\mathbf{C}}^{(s)} = -(\hat{\mathbf{C}}^{(s)})^{\top}$, the interconnection contribution $\hat{\mathbf{B}}\hat{\mathbf{C}}^{(s)}\hat{\mathbf{B}}^{\top}$ is skew-symmetric as well, so Eq. (9) is again a port-Hamiltonian system with additive storage H_{total}.

Theorem 3.1 (Power-preserving interconnection)**.** *Consider s port-Hamiltonian subsystems with coupling-port variables $(\hat{u}_i, \hat{y}_i)$ and total energy $H_{total} = \sum_{i=1}^{s} H_i$. If the coupling constraint is $\hat{u} = -\hat{\mathbf{C}}^{(s)}\hat{y}$ with $\hat{\mathbf{C}}^{(s)} = -(\hat{\mathbf{C}}^{(s)})^{\top}$, then the coupling injects zero net power:*

$$\sum_{i=1}^{s} \hat{u}_i^{\top} \hat{y}_i \;=\; \hat{y}^{\top}\hat{u} \;=\; -\hat{y}^{\top}\hat{\mathbf{C}}^{(s)}\,\hat{y} \;=\; 0. \tag{10}$$

Consequently, the energy balance of the composed system depends only on dissipation and the external *ports (motor torques / control), not on the interconnection.*

Intuitively, each subsystem acts like a local energy reservoir: external ports can inject or dissipate energy, while the skew interconnection only redistributes energy between subsystems and never creates it.

Theorem 3.1 is not new pH composition theory: the power-preserving interconnection of port-Hamiltonian systems through a skew (Dirac) structure is classical (Cervera et al., 2007; Van Der Schaft & Jeltsema, 2014), and compositional PHNNs already learn subsystems that compose under this principle (Neary & Topcu, 2023). We state it here to make explicit that C-PHAST's typed, neural, deployment-time instantiation *preserves* that classical guarantee rather than to claim it as the paper's core novelty. The novelty targeted by C-PHAST is not skew symmetry itself but skew symmetry combined with q-only state lifting and target-specification redeployment; the corresponding finite-step energy-balance-defect, deployment-transfer, and consequence-reliability bounds are established in Section 3.8 and Appendices B–D, under explicit hypotheses whose constants are instantiated for the KNOWN-regime examples and empirically calibrated for the trained deployments.

Candidate futures as deployed coupling graphs. The candidate-future intuition in Figure 1 can now be written directly in this notation. All three futures share the same observed scene, the same local arm/base asset library, and the same q-only observation interface; only the active support/contact graph changes. Since only the graph varies here, we abbreviate the corresponding deployed operator for future k as $\hat{\mathbf{C}}^{(G_k)}$ in this example. Writing G_1, G_2, G_3 for the efficient single-arm reach, the stable multi-arm brace, and the unsafe under-braced reach, the deployed coupling law is always

$$\hat{u} = -\hat{\mathbf{C}}^{(G_k)}\hat{y}, \qquad k \in \{1, 2, 3\},$$

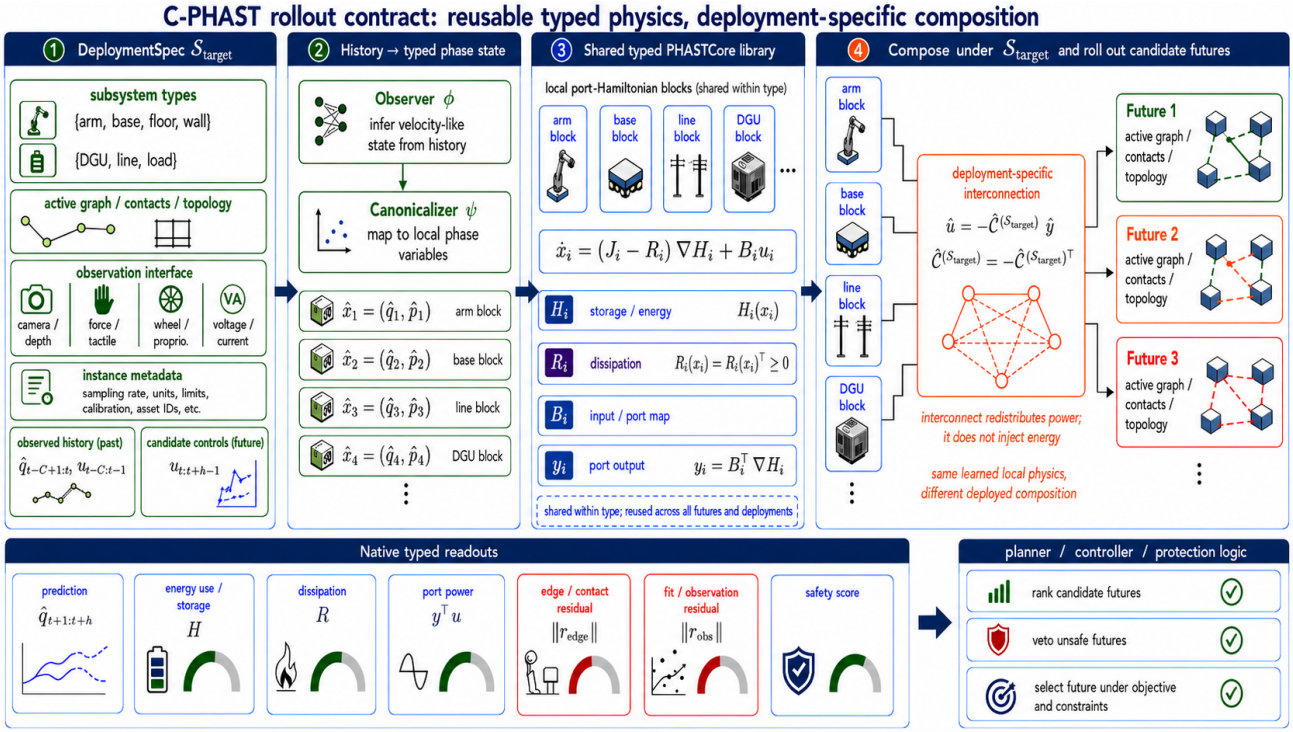

Figure 4. **C-PHAST rollout contract: reusable typed physics, deployment-specific composition.** Figures 1 and 2 show the robotics and microgrid motivations; this figure writes the shared computational contract, and Figure 10 instantiates the same contract in Isaac Lab. A deployment specification $\mathcal{S}_{\text{target}}$ declares which physical parts, cells, or assets are present; who is coupled to whom; which measurements are available; what instance metadata is known; the past observations; and the candidate future controls. The front end maps history into typed phase states, a shared typed PHASTCore library supplies local port-Hamiltonian storage, dissipation, input/port maps, and port outputs, and the deployed skew interconnection recomposes those blocks under $\mathcal{S}_{\text{target}}$ without injecting net energy. The same local physics can therefore be rolled out under different deployed compositions, producing candidate futures with typed readouts such as predicted trajectory, energy/storage, dissipation, port power, edge/contact residuals, observation-fit residuals, and safety scores. These readouts are consumed by planner, controller, or protection logic to rank, veto, or select futures under task-specific objectives and constraints.

but with different skew-symmetric deployed operators (with task-dependent scalar entries c_{ij}):

$$\hat{\mathbf{C}}^{(G_1)} = 0, \qquad \hat{\mathbf{C}}^{(G_2)} = \begin{bmatrix} 0 & c_{12} & 0 \\ -c_{12} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad \hat{\mathbf{C}}^{(G_3)} = \begin{bmatrix} 0 & c_{12} & c_{13} \\ -c_{12} & 0 & c_{23} \\ -c_{13} & -c_{23} & 0 \end{bmatrix}.$$

In the efficient but risky future, no arm-to-arm support coordination is required, so the coupling vanishes. In the stable bracing future, only arms 1 and 2 exchange power through one coordination edge. In the unsafe reach, all three physical parts may appear in the scene but the active support graph is insufficient or mismatched, and the residual channels expose that the candidate is near or beyond the physical envelope. The local arm dynamics are unchanged across all three futures; only the deployed interconnection operator and active contact/support graph are rebuilt. Figure 4 lifts this toy example into the full rollout contract: deployment specification, history-to-phase inference, shared typed PHASTCore blocks, deployment-specific interconnection, and native typed readouts for downstream decision logic.

Homogeneous count-transfer coupling parameterization. The toy example above fixed the deployment count, typed layout, observation interface, and instance metadata and varied only the coordination graph. The homogeneous count-transfer specialization instead indexes the deployed operator only by the subsystem count s. In our implementation the coupling parameters θ_C are the entries of an unconstrained matrix $\mathbf{U}_{\theta_C} \in \mathbb{R}^{s_{\max} \times s_{\max}}$, from which we construct the homogeneous deployed coupling matrix

$$\hat{\mathbf{C}}^{(s)} \quad \text{for each deployed subsystem count } s.$$

Specifically,

$$\hat{\mathbf{C}}^{(s)} = \frac{1}{\sqrt{s}}\big((\mathbf{U}_{\theta_C})_{1:s,1:s} - (\mathbf{U}_{\theta_C})_{1:s,1:s}^{\top}\big), \tag{11}$$

which guarantees $\hat{\mathbf{C}}^{(s)} = -(\hat{\mathbf{C}}^{(s)})^{\top}$ by construction and allows evaluation at any $s \le s_{\max}$ without changing parameters. Training at a source count s_0, however, supplies gradients only to the top-left $s_0 \times s_0$ block, so slicing to $s > s_0$ *evaluates* at the larger count but does not *learn* the new coupling entries; top-left slicing is moreover not permutation-equivariant across otherwise-equivalent repeated blocks. We therefore distinguish *local-block count transfer*—the shared per-block dynamics carrying over to more blocks, which the KNOWN/PARTIAL coupling rows support strongly—from *learned-coupling count extrapolation*—predicting genuinely new inter-block couplings—for which a size-equivariant edge kernel $C_{ij} = f_\theta(\text{type}_i, \text{type}_j, \text{edge}_{ij})$ (with $C_{ji} = -C_{ij}$) would be the principled parameterization; the present dense-slice evidence for the latter is correspondingly weaker. We use the $1/\sqrt{s}$ normalization by default and ablate alternative normalizations in Appendix G.8.

For chain-structured systems, a nearest-neighbor template with $O(1)$ parameters is also available (Appendix F). Each subsystem exposes velocity port output $\hat{y}_i = \partial H_i/\partial p_i$; for multi-DOF subsystems we use $\hat{\mathbf{C}}^{(s)} \otimes I_d$ so the coupling remains defined at the subsystem level while the parameterization stays at $O(s^2)$ (dense) or $O(1)$ (chain).

Port-layered dynamics: internal, external, and domain-specific. The previous subsection defined the generic pH interconnection. We now add back the remaining inputs needed by the actual benchmarks: local constitutive or actuation ports and, when necessary, an explicit domain-specific forcing term. The key modeling split is: local subsystem physics stays inside each typed block, generic composition enters only through the skew interconnection, and any benchmark-specific forcing that does not fit that generic template is carried by a separate domain port. Formally, let u_i^{loc} denote any local external or constitutive input exposed by subsystem i, let $\hat{y}_i := \partial H_i/\partial p_i$ denote its coupling-port output, and let $u_i^{\text{dom}}(x)$ denote an optional domain-specific input written in the same power coordinates as $\hat{y}_i$. Writing $\hat{C}_{ij}$ for the (i,j) entry of the deployed coupling operator, $\hat{\mathbf{C}}^{(s)}$ in homogeneous settings and, more generally, $\hat{\mathbf{C}}^{(\mathcal{S}_{\text{target}})}$, the total port input seen by subsystem i is therefore

$$u_i^{\text{tot}} = u_i^{\text{int}} + u_i^{\text{ext}}, \qquad u_i^{\text{int}} := \hat{u}_i, \qquad u_i^{\text{ext}} := u_i^{\text{loc}} + u_i^{\text{dom}}(x),$$

where $\hat{u}_i$ is the internal component selected by the skew interconnection law in Eq. (8). With $v_i := \partial H_i/\partial p_i$, the per-subsystem dynamics become

$$\dot{x}_i = \underbrace{(J_i - R_i)\nabla H_i + \mathbf{B}_i u_i^{\text{loc}}}_{\text{local pH core + ports}} - \underbrace{\mathbf{B}_i \textstyle\sum_j \hat{C}_{ij}\hat{y}_j}_{\text{interconnection}} + \underbrace{\mathbf{B}_i u_i^{\text{dom}}(x)}_{\text{domain port}} \tag{12}$$

Intuitively, Eq. (12) says: "share the blue local law, resize the orange interconnection, and keep any green benchmark-specific forcing explicit rather than hiding it inside the coupling." The local layer contains the per-subsystem Hamiltonian, dissipation, and constitutive ports such as gravity or tunnel-diode nonlinearities, and is shared via weight tying across all subsystems of the same type. The interconnection is power-preserving by construction ($\hat{y}^{\top}\hat{\mathbf{C}}^{(\mathcal{S}_{\text{target}})}\hat{y} = 0$ for any deployed skew operator) and is re-instantiated from the relevant components of $\mathcal{S}_{\text{target}}$. The domain port is optional and carries benchmark-specific forcing that should remain explicit rather than be absorbed into the generic skew interconnection. Examples include converter/load forcing in the microgrid. By contrast, ordinary mechanical control torques are treated as local external ports of each typed block rather than as a separate green domain input. This decomposition directly yields the storage balance

$$\frac{dH_{\text{total}}}{dt} = \underbrace{-\sum_i v_i^{\top} R_i v_i}_{\le 0 \text{ (dissipation)}} + \underbrace{\sum_i v_i^{\top} u_i^{\text{loc}}}_{\text{local ports}} + \underbrace{0}_{\hat{y}^{\top}\hat{\mathbf{C}}^{(\mathcal{S}_{\text{target}})}\hat{y}} + \underbrace{\sum_i v_i^{\top} u_i^{\text{dom}}(x)}_{\text{domain}} \tag{13}$$

In the unforced forecasting setting, $u_i^{\text{loc}} = 0$, so only dissipation and any benchmark-specific domain input contribute to the continuous-time storage balance; the generic interconnection itself contributes zero. The finite-step counterpart of this continuous balance is made precise in Appendix B: with second-order accounting of the boundary power terms, one structure-preserving step satisfies the balance up to an $\mathcal{O}(\Delta t^3)$ defect plus a first-order observer/chart term under q-only sensing (Theorem B.1), a bound we verify directly against the logged energy-balance residual. The accumulated horizon-h error is controlled by the deployment-transfer bound of Section 3.8 (Theorem C.2), under a contraction hypothesis on the deployment's operating region (interval-certified for the linear KNOWN deployments, measured otherwise); the empirical rollout results below exhibit that horizon-uniform behavior in the contractive regime.

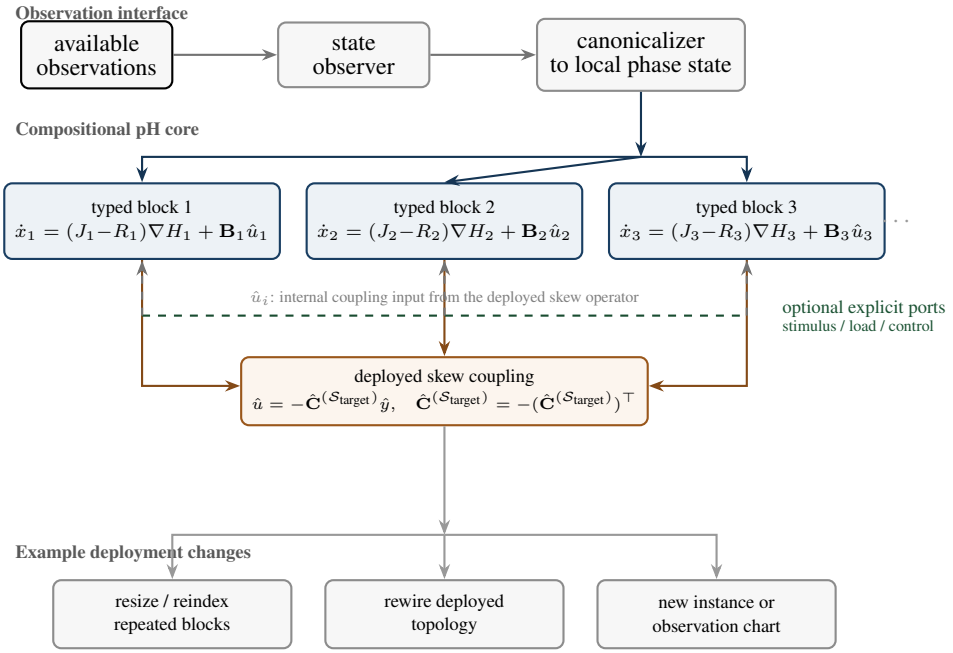

Figure 5. **The C-PHAST contract.** Every benchmark instantiates the same observation-to-composition schema: available measurements are mapped to local pH states, typed local blocks carry reusable storage and port structure, and the deployed skew interconnection is rebuilt from the target specification. Blue marks reusable local physics, orange marks deployed power-preserving coupling, and green marks explicit domain ports such as stimulus, load, or external forcing. The block equations show only the generic internal coupling interface $\mathbf{B}_i \hat{u}_i$; benchmark-specific forcing remains visible through separate domain ports rather than being absorbed into generic coupling. The bottom row illustrates common deployment changes, not an exhaustive mode list: C-PHAST can resize repeated blocks, rewire topology, reuse a typed layout on a new instance, or change the observation chart. The KNOWN, PARTIAL, and LAW-UNKNOWN regimes change how much of this contract is specified versus learned.

Typed residual channels. The same decomposition also defines diagnostics. Given a rollout and a reference trajectory when available, C-PHAST can report channel-wise residuals rather than only a scalar prediction error:

$$r_{\text{obs}}, \qquad r_{\text{block}}, \qquad r_{\text{energy}}, \qquad r_{\text{port}}, \qquad r_{\text{edge}}.$$

Here r_{obs} measures deployed observation mismatch, r_{block} measures typed local-state mismatch, r_{energy} measures violation of the storage balance in Eq. (13), r_{port} measures local or domain-port work mismatch, and r_{edge} measures interconnection or edge-power mismatch. These residuals are not extra learned heads; they are bookkeeping consequences of the typed pH rollout. They become useful when the question is not only whether the model predicts well, but which compositional object or channel stopped matching the deployed system. Before the algorithms, Figure 5 and the next two tables answer the recurring deployment question in visual form: what stays local and reusable, what is rebuilt as deployed wiring, and what enters or is read out explicitly. Table 5 gives the port-level split for each benchmark domain, and Table 6 summarizes the benchmark-specific transfer axes. A fourth layer for Energy–Casimir control ports is developed later in Section 5.

Block granularity. There are two useful levels of decomposition. A *pH rollout block* is a local subsystem advanced by a PHASTCore state update, with local coordinates, storage, damping, ports, and shared parameters. An *interface block* owns observed streams, port readouts, consequence channels, energy/accounting channels, or planner-facing factors. In the arm, legged, FHN, and microgrid benchmarks, the interface blocks are backed by pH rollout blocks. The Spot result uses the same typed interface at real-robot scale, with interface blocks for base motion, leg support/contact, arm motion/contact, camera/depth sensing, and battery/power accounting. Promoting those Spot interface blocks into a full PHASTCore rollout hierarchy is the natural closed-loop hardware direction.

Table 6 is the main-text contract map. The appendix mirrors the same rows benchmark by benchmark, giving the protocols, ablations, diagnostics, and controller-facing extensions without changing the contract stated here.

Table 5. **Port decomposition by domain.** This table is the concrete legend for the paper's color grammar. Blue is the local physics that can be reused, orange is the deployed exchange pattern, and green is explicit input or forcing that should not be hidden inside generic coupling.

System	Local Block	Block Exchange	Explicit Input / Port
Planar arm (N=2–8 joints)	One joint: $V{=}g(1{-}\cos\theta)$, $M{=}I$, damping b_i	learned skew coupling across joints (Cayley)	joint torque τ
Legged robots (q-only)	MuJoCo: one 2-DOF leg; Isaac: one 3-DOF leg (HAA/HFE/KFE)	learned skew exchange across deployed leg blocks (Cayley)	motor torque $\mathbf{B}(q)\tau$
FHN circuit (N excitable cells)	Excitable cell (V, W): capacitor/inductor storage; cubic/linear constitutive terms	diffusive voltage coupling on target graph: $g\, L_{\mathcal{S}} V$	stimulus/source current I
DC microgrid (N DGUs + M lines)	DGU block: capacitor, inductor, resistor; line block: inductor, resistor	incidence matrix B_{inc} routes line currents and voltage differences	converter control u_i; load current I_{Li}

Table 6. **Hierarchy used by each benchmark.** Figure 5 gives the invariant architecture; this table says what changes from domain to domain. Read each row as the hierarchy used in this paper: what one pH rollout block means, what the interface exposes, and what evidence role the benchmark plays.

Domain	Hierarchy	pH rollout block	Interface / readout level	Evidence role
Planar arm	arm $\rightarrow$ N joints	one 1-DOF joint	joint torque, q-only rollout, energy and coupling diagnostics	clean mechanical count-scaling check
Legged robots	robot $\rightarrow$ repeated legs; Isaac uses four HAA/HFE/KFE legs	MuJoCo: one 2-DOF leg; Isaac: one 3-DOF leg, floating base excluded	q-only leg state, motor torque, support/contact diagnostics	leg-count and cross-robot transfer
FHN circuits	chain of repeated excitable cells	one voltage/recovery cell	source current, cell readout, diffusive graph port	non-robot repeated-block transfer check
DC microgrid	network $\rightarrow$ DGU nodes and line edges	typed DGU block and typed line block	voltage/current readouts, load and converter-control ports	heterogeneous typed-block recomposition, reconfiguration, and sparse sensing
Spot with arm	robot $\rightarrow$ base, leg support/contact, arm, camera/depth, battery/power interface blocks	interface-block hierarchy now; PHASTCore hardware rollout in follow-up	telemetry histories, candidate descriptors, typed consequences, and action cards	real-robot typed consequence forecasting

3.4. Local pH Coordinates Before PHASTCore

The deployment chart introduced above maps raw measurements into model-facing coordinates. PHASTCore expects one additional alignment step: it advances local pH states, not raw telemetry. The chart says which local coordinate patch of the deployed system the model is moving in; the observer or canonicalizer says which storage and port variables should be advanced inside that patch. For FHN and microgrids, the benchmark declares these local variables directly, such as repeated excitable-cell states or DGU/line storage and port variables after the chosen normalization. For mechanical systems, the local pH state is canonical phase space (q, p).

Mechanical canonicalization. Many robot pipelines provide joint angles q, while velocities $\dot{q}$ may be noisy or unavailable; Section 3.6 later gives the causal observer used in that deployment setting. The only mechanical fact needed here is that momentum need not equal velocity: the canonical momentum is $p = M(q)\dot{q}$, where $M(q)$ is the configuration-dependent mass matrix. This moves the state from $(q, \dot{q})$ coordinates, where mass and velocity estimation are tied to the particular embodiment, to canonical phase-space coordinates (q, p) in which the pH structure takes a standard form:

$$\begin{pmatrix} \dot{q} \\ \dot{p} \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix}}_{J_{\text{canonical}}} \nabla H(q,p) + \text{dissipation} + \text{control} \tag{14}$$

The key point is that the symplectic structure $J_{\text{canonical}}$ is independent of robot geometry: a 2-joint arm and a 12-joint quadruped share the same canonical pH form once written in (q, p) coordinates. Across fixed-layout robot instances, target metadata Ξ_{target} then enters as a local response mismatch rather than as a topology change: mass, geometry, actuator response, and friction change the map from model-coordinate q histories to (q, p), even when the typed layout and deployed coupling graph are unchanged.

Why this matters for composition. The preceding coordinate step is not cosmetic: a shared block only makes sense if the state can be written in local variables whose stored energy is mostly local. The decoupling theory of (Ehrhardt et al., 2026) shows that a monolithic pHODE can be split into subsystems when a coordinate transformation exposes an additive Hamiltonian, $H(x) = \sum_i H_i(w_i)$, over local coordinate blocks w_i. C-PHAST uses this as a modeling bias rather than as an exact guarantee: choose local coordinates that expose repeated storage terms, learn a shared local block in those coordinates, and leave remaining power exchange to the interconnection and domain ports. In robotics this means that approximately block-local mechanical energy makes shared joint or limb blocks meaningful; in FHN and microgrids the repeated excitable cells, DGUs, and lines supply the local typed blocks directly.

The bridge used by the rest of the method is therefore

raw observations $\rightarrow$ normalized/wrapped coordinates $\rightarrow$ local pH state $\rightarrow$ shared PHASTCore update.

The local learnable block is specified next; the practical q-only realization of the observer and canonicalizer is deferred to Section 3.6.

3.5. PHASTCore: Dissipative Mechanics + Strang Splitting

PHAST blocks and knowledge regimes. Once each subsystem has been represented in local pH coordinates, C-PHAST applies the same PHASTCore update to every repeated block. Each per-subsystem dynamics block is an explicit port-Hamiltonian step with learnable potential $V_{\theta_V}(q)$, mass $M_{\theta_M}(q)$, and damping $D_{\theta_D}(q)$ modules. The same block template supports three *knowledge regimes*: KNOWN instantiates specified physics modules on the supplied target layout, PARTIAL augments typed modules with learnable residuals or observation interfaces, and LAW-UNKNOWN keeps the declared subsystem layout but learns the local dynamics and coupling subject to physical constraints. These regime labels are domain-grounded rather than perfectly uniform: in robotics, the maintained PARTIAL regime is defined by what the asset and IO stack actually expose, including grouped articulation, observation/action semantics, and coarse instance metadata, whereas in the microgrid benchmark the corresponding structure includes explicit typed interconnection and local ports. Across our experiments, we use the same PHASTCore discretization in all regimes; the regime only determines which components are given vs learned. This is also where weight sharing enters concretely: the same local update rule is reused across all instantiated subsystems of a given type.

In the mechanical setting with state $x = (q, p)$ (configuration q and momentum p, each in $\mathbb{R}^d$ for a d-DOF subsystem), we use the mechanical Hamiltonian

$$H_{\theta_{\text{loc}}}(q,p) = V_{\theta_V}(q) + \tfrac{1}{2} p^\top M_{\theta_M}(q)^{-1} p, \tag{15}$$

with $M_{\theta_M}(q) \succ 0$ parameterized as $M_{\theta_M}(q) = L_{\theta_M}(q) L_{\theta_M}(q)^\top + \varepsilon I$ ($\varepsilon > 0$) so positive definiteness and a uniform eigenvalue floor $\lambda_{\min}(M_{\theta_M}) \geq \varepsilon$ are architectural; bounded conditioning additionally requires a cap or parameterization controlling $\lambda_{\max}(M_{\theta_M})$. Equation (15) is a *mechanical*, not a standard separable, Hamiltonian once M_{θ_M} depends on q: the second-order explicit-leapfrog result of Theorem B.1 is certified for the constant, scalar, or blockwise-constant mass deployments evaluated here, while a strongly configuration-dependent $M(q)$ requires a generalized nonseparable

integrator and lies outside the current finite-step certificate. We then choose a dissipation matrix that acts on momenta, $R_{\theta_D} = \mathrm{diag}(0, D_{\theta_D}(q))$ with $D_{\theta_D}(q) \succeq 0$. The damping block is not just a stabilizing term: because it is PSD by construction, its low-rank factors can be exposed as typed dissipative modes. When the optional partial-flag parameterization is active, these modes become ordered spectral coordinates that can be logged as a typed damping diagnostic. With total port input $u_i^{\mathrm{tot}} = u_i^{\mathrm{loc}} + \hat{u}_i + u_i^{\mathrm{dom}}$, this yields the standard passivity identity

$$\frac{dH_{\theta_{\mathrm{loc}}}}{dt} = -v^\top D_{\theta_D}(q)\, v + (y_i^{\mathrm{port}})^\top u_i^{\mathrm{tot}} \le (y_i^{\mathrm{port}})^\top u_i^{\mathrm{tot}}, \qquad v := \frac{\partial H_{\theta_{\mathrm{loc}}}}{\partial p} = M_{\theta_M}(q)^{-1} p. \tag{16}$$

When all ports are closed, this reduces to monotone energy dissipation. Figure 6 shows this single-block interface explicitly: the same PHASTCore block exposes local/control, internal coupling, and optional domain ports.

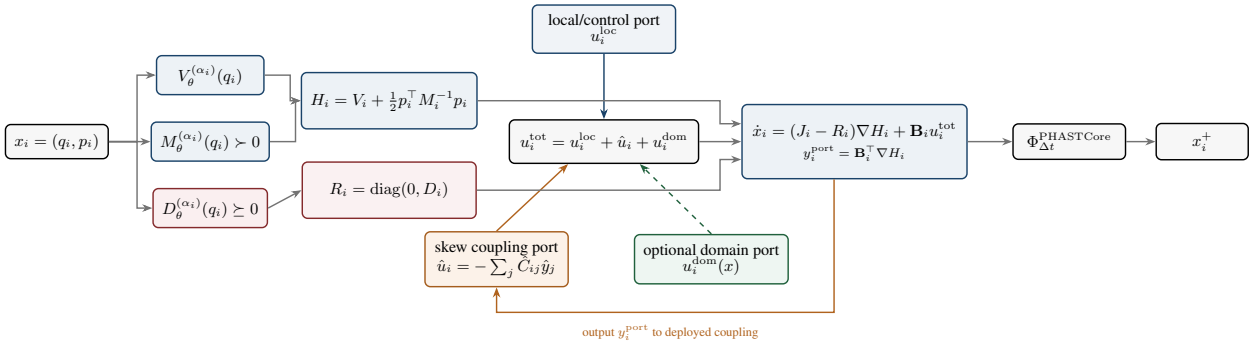

Figure 6. **One reusable PHASTCore block with ports.** For a typed subsystem α_i, the local modules $V_\theta^{(\alpha_i)}$, $M_\theta^{(\alpha_i)}$, and $D_\theta^{(\alpha_i)} \succeq 0$ assemble into a mechanical port-Hamiltonian block with port matrix $\mathbf{B}_i$. Local/control inputs enter through u_i^{loc}, C-PHAST composes blocks through the orange skew-coupling port $\hat{u}_i$, and optional benchmark-specific forcing enters through u_i^{dom}. The continuous-time storage balance is $\dot{H}_i = -v_i^\top D_i v_i + (y_i^{\mathrm{port}})^\top (u_i^{\mathrm{loc}} + \hat{u}_i + u_i^{\mathrm{dom}})$. This is Eq. (16) instantiated inside the block-level interface of Eq. (12). Under skew-symmetric coupling, the orange power terms cancel after summing over blocks, while the blue and green ports remain explicit. Algorithm 1 calls this block in its blue PHASTCore stage.

Damping and integration. Dissipation $D(q) \succeq 0$ is parameterized as a Householder-style low-rank expansion (Appendix F), guaranteeing positive semi-definiteness by construction. Optionally bounding $\|D\|$ prevents damping from absorbing prediction errors meant for V and M. This parameterization is also useful conceptually: it separates damping into a small number of dissipative directions and strengths, rather than treating D as an unstructured PSD matrix. For a local velocity v, the ordered low-rank specialization with rank r_D decomposes dissipated power as

$$v^\top D(q) v = d_0 \|v\|^2 + \sum_{\ell=1}^{r_D} \beta_\ell(q) \big(k_\ell(q)^\top v\big)^2, \tag{17}$$

so each learned mode identifies a direction in local phase motion where energy is removed. When the directions are constrained to be orthonormal and the strengths are ordered, the damping sector becomes an identifiable modal coordinate system up to sign. This ordered specialization gives a direct diagnostic interface: changes that mostly alter friction, contact loss, actuator damping, or electrical dissipation can be read as motion of dissipative modes rather than as arbitrary movement of the conservative Hamiltonian core. When this option is enabled, we use the spectrum as a conservative warning channel rather than as the full physical critic.

Structure-preserving integration (Strang splitting). To advance dissipative dynamics we use Strang splitting:

$$\Phi_{\Delta t} = \Phi_D^{\Delta t/2} \circ \Phi_H^{\Delta t} \circ \Phi_D^{\Delta t/2}, \tag{18}$$

where $\Phi_H^{\Delta t}$ is a conservative (symplectic) update and $\Phi_D^{\Delta t/2}$ is a half-step of damping. The per-subsystem step $\Phi_{\Delta t}$ uses leapfrog (Störmer–Verlet) for the conservative core (Appendix F). The dissipative half-step integrates $\dot{\boldsymbol{p}} = -A_D \boldsymbol{p}$ with $A_D = D(q)M(q)^{-1}$ and position frozen, whose exact flow is $\exp(-A_D\tau)\boldsymbol{p}$ ($\tau = \Delta t/2$). We use the Cayley (trapezoidal / $[1/1]$-Padé) step

$$\Phi_D^\tau : \ \boldsymbol{p} \longmapsto (I + c\, A_D)^{-1}(I - c\, A_D)\, \boldsymbol{p}, \qquad A_D = D(q)M(q)^{-1}, \quad c = \tfrac{\tau}{2} = \tfrac{\Delta t}{4}, \tag{19}$$

which matches $\exp(-A_D\tau)$ to $\mathcal{O}(\tau^3)$—so the full split step is genuinely second order—and is *unconditionally contractive in the kinetic-energy metric*. Because $A_D = DM^{-1}$ is generally nonnormal, a spectral-radius argument does not by itself give a norm contraction; the correct statement is that A_D is self-adjoint and positive semidefinite in the inverse-mass inner product $\langle \boldsymbol{p}, \boldsymbol{q}\rangle_{M^{-1}} = \boldsymbol{p}^\top M^{-1}\boldsymbol{q}$ (indeed $M^{-1}A_D = M^{-1}DM^{-1}$ is symmetric and, since $D \succeq 0$, positive semidefinite). It is therefore orthogonally diagonalizable in that metric with real eigenvalues $\lambda \geq 0$, and its Cayley transform contracts the induced norm, $|(1 - c\lambda)/(1 + c\lambda)| \leq 1$ for every $\lambda \geq 0$ and every Δt, so the half-step is non-expansive in the kinetic energy $\boldsymbol{p}^\top M^{-1}\boldsymbol{p}$ and never injects energy (A-stability). The step is A-stable but not L-stable: as $\tau\lambda \to \infty$ the amplification factor tends to -1, so very stiff modes are not strongly damped—the momentum can flip sign rather than decay—while the energy stays non-increasing, which is all the finite-step defect requires. This is the improvement over the second-order Taylor truncation $I - \tau A_D + \frac{1}{2}\tau^2 A_D^2$, which amplifies once $\tau\,\lambda_{\max}(A_D) > 2$; and over a forward-Euler half, which is only first order. The step needs the small per-block solve $(I + cA_D)^{-1}$; the operator A_D is assembled from the damping/mass matrix–vector products, so state-dependent and learned $D(q)$ are covered. Algorithms 1 and 2 expose this deployed update in two layers: first the homogeneous mechanical latent kernel, then the q-only predictor that wraps it. Here $\bar{\boldsymbol{u}}^{\mathrm{loc}} \in \mathbb{R}^N$ denotes the stacked local/control port inputs and $\bar{\boldsymbol{u}}^{\mathrm{dom}} \in \mathbb{R}^N$ denotes optional domain-port inputs (zero when absent). The matrix M_{eff}^{-1} is not a second learned Hamiltonian mass module: $M_{\theta_M}(q)$ appears in the local kinetic energy in Eq. (15), while M_{eff}^{-1} is the fixed inverse-mass metric used only by the Cayley coupling step to convert deployed skew exchange into a momentum update. In the current experiments this metric is the identity or the specified effective mass used by the deployment/canonicalizer. For the finite-step certificate of Theorem B.1 to model the same coupling law as the continuous port output $\hat{\boldsymbol{y}} = M_{\theta_M}(q)^{-1}\boldsymbol{p}$, the effective metric is taken to match the Hamiltonian mass, $M_{\mathrm{eff}} = M_{\theta_M}$, in the constant, scalar, or blockwise-scalar regime where the two coincide; a distinct $M_{\mathrm{eff}} \neq M_{\theta_M}$ would define a different modeled coupling law and falls outside the certified set. For readability, define the Cayley half-step operator

$$\mathcal{C}_{\Delta t/2}(A)\boldsymbol{p} := \left(I + \tfrac{\Delta t}{4}A\right)^{-1}\left(I - \tfrac{\Delta t}{4}A\right)\boldsymbol{p}, \qquad A := M_{\mathrm{eff}}^{-1}\hat{\mathbf{C}}^{(s)}. \tag{20}$$

The generator $A = M_{\mathrm{eff}}^{-1}\hat{\mathbf{C}}^{(s)}$ is skew in the M_{eff}-inner product, so $\mathcal{C}_{\Delta t/2}$ preserves $\boldsymbol{p}^\top M_{\mathrm{eff}}\boldsymbol{p}$. This coincides with the kinetic form $\boldsymbol{p}^\top M_{\mathrm{eff}}^{-1}\boldsymbol{p}$ exactly when M_{eff} is a scalar multiple of the identity, or blockwise scalar on every coupled subspace, as in our deployments—a diagonal but *non*-scalar mass need not commute with a dense skew $\hat{\mathbf{C}}^{(s)}$, so "diagonal mass" alone is insufficient. For a general noncommuting effective mass the kinetic form is preserved by the momentum generator $A_p = \hat{\mathbf{C}}^{(s)}M_{\mathrm{eff}}^{-1}$ (skew in the M_{eff}^{-1}-metric), or equivalently by applying the Cayley map in velocity coordinates ($\boldsymbol{v} = M_{\mathrm{eff}}^{-1}\boldsymbol{p}$, update $\boldsymbol{v}$, then $\boldsymbol{p}^+ = M_{\mathrm{eff}}\boldsymbol{v}^+$). This step is metric-preserving, not canonically symplectic: the momentum-only map $(q, \boldsymbol{p}) \mapsto (q, Q\boldsymbol{p})$ has Jacobian $\mathrm{diag}(I, Q)$ and is symplectic only when $Q = I$. Algorithm 1 assumes that the observation front end has already produced phase-space subsystem states. It then applies the same logical sequence introduced above: local/control ports, optional domain ports, power-preserving coupling, shared PHASTCore updates, and the symmetric closing half-steps.

Algorithm 1 Latent C-PHAST transition: fixed local blocks, explicit ports, deployed coupling

Require: Subsystem states $\{(q_i, p_i)\}_{i=1}^{s}$, stacked local/control inputs $\bar{\boldsymbol{u}}^{\mathrm{loc}} \in \mathbb{R}^N$, optional domain-port inputs $\bar{\boldsymbol{u}}^{\mathrm{dom}} \in \mathbb{R}^N$ (zero if absent), deployed skew coupling $\hat{\mathbf{C}}^{(s)} = -(\hat{\mathbf{C}}^{(s)})^\top$ from Eq. (8), PHASTCore step $\Phi_{\Delta t}$ from Fig. 6 and Eq. (18), inverse mass proxy M_{eff}^{-1}, time step Δt

Ensure: Updated subsystem states $\{(q_i, p_i)\}_{i=1}^{s}$

State assembly — mechanical latent state from Eq. (15)

$\boldsymbol{p} \leftarrow [p_1, \ldots, p_s]$, $\quad A \leftarrow M_{\mathrm{eff}}^{-1} \hat{\mathbf{C}}^{(s)}$ {Eq. (20)}

Local/control port half-kick — blue ports in Eq. (12); storage balance Eq. (16)

$\boldsymbol{p} \leftarrow \boldsymbol{p} + \frac{\Delta t}{2} \bar{\boldsymbol{u}}^{\mathrm{loc}}$ {commanded/controller work}

Optional domain-port half-kick — green ports in Eq. (12); e.g., contact/support/load forcing

$\boldsymbol{p} \leftarrow \boldsymbol{p} + \frac{\Delta t}{2} \bar{\boldsymbol{u}}^{\mathrm{dom}}$ {zero when no green port is declared}

Power-preserving coupling half-step — skew law Eq. (8); Cayley map Eq. (20)

$\boldsymbol{p} \leftarrow \mathcal{C}_{\Delta t/2}(A)\boldsymbol{p}$

$(p_1, \ldots, p_s) \leftarrow \boldsymbol{p}$

Shared local PHASTCore update — single-block interface Fig. 6; Strang split Eq. (18)

for $i = 1$ to s **do**

  $(q_i, p_i) \leftarrow \Phi_{\Delta t}(q_i, p_i)$

end for

Power-preserving coupling closing half-step — symmetric Cayley step from Eq. (20)

$\boldsymbol{p} \leftarrow [p_1, \ldots, p_s]$

$\boldsymbol{p} \leftarrow \mathcal{C}_{\Delta t/2}(A)\boldsymbol{p}$

Optional domain-port closing half-kick — reverse-order close of the green half-step

$\boldsymbol{p} \leftarrow \boldsymbol{p} + \frac{\Delta t}{2} \bar{\boldsymbol{u}}^{\mathrm{dom}}$

Local/control port closing half-kick — reverse-order close of the blue half-step

$\boldsymbol{p} \leftarrow \boldsymbol{p} + \frac{\Delta t}{2} \bar{\boldsymbol{u}}^{\mathrm{loc}}$

$(p_1, \ldots, p_s) \leftarrow \boldsymbol{p}$

return updated $\{(q_i, p_i)\}_{i=1}^{s}$

Algorithm 1 displays the certified zero-order-held, fixed-coupling path: one generator A and one $\bar{\boldsymbol{u}}^{\mathrm{dom}}$ are reused in both half-steps. A state-dependent domain port or a configuration-dependent coupling requires symmetric endpoint reevaluation (A_{open}=$A(q_k)$, A_{close}=$A(q_{k+1})$, and likewise for $\boldsymbol{u}^{\mathrm{dom}}$) and, when evaluated only at the opening state, falls outside the second-order certificate of Theorem B.1.

3.6. Position-Only Observation (Q-Only Setting)

In many robotic deployments, sensors measure only joint angles q_t (e.g., from encoders), while velocities $\dot{q}_t$ and momenta p_t must be inferred rather than directly observed.

Deployed rollout pipeline. At time t, the deployed model receives a raw position-like measurement window $\tilde{q}_{t-L_h+1:t}$, possibly corrupted by sensing noise. Following the state pipeline above, the deployment chart first maps this window into model coordinates, $z_\tau = \chi_{\mathrm{target}}(\tilde{q}_\tau)$ for $\tau = t - L_h + 1, \ldots, t$. A history-to-state map then lifts the model-coordinate history into the latent pH state:

$$\hat{x}_t = \Phi_{\phi,\psi}(z_{t-L_h+1:t}, u_{t-L_h:t-1}), \qquad \hat{z}_t = \Pi_q(\hat{x}_t), \tag{21}$$

Here $z_{t-L_h+1:t}$ denotes the inclusive window $(z_{t-L_h+1}, \ldots, z_t)$, and $\hat{z}_t$ is the model-coordinate readout implied by the inferred latent state. The map $\Phi_{\phi,\psi}$ is composed of an observer o_ϕ (velocity extraction) and optional mass-based canonicalizer C_ψ. For the fixed deployment specification active during this rollout, denote the resulting one-step transition by

$$\hat{x}_{t+1} = G_\theta^{(\mathcal{S}_{\mathrm{target}})}(\hat{x}_t, u_t), \qquad \hat{q}_{t+1}^{\mathrm{obs}} = \chi_{\mathrm{target}}^{-1}(\Pi_q(\hat{x}_{t+1})), \tag{22}$$

where $G_\theta^{(\mathcal{S}_{\text{target}})}$ abbreviates the executable transition assembled by the staged blocks of Algorithm 2: normalize or wrap observations, lift the resulting history to a latent pH state, instantiate the deployment composition, advance latent compositional dynamics, and read out observed coordinates. Inside that latent transition, Algorithm 1 applies the two power-preserving Cayley coupling stages around the shared local PHASTCore update. The projection Π_q reads the model-coordinate position component before the inverse chart decodes it to observation coordinates. This q-only wrapper is the robotics execution path corresponding to Eq. (22). Non-mechanical domains instantiate the same operator contract with their own benchmark-specific local/domain ports.

Algorithm 2 Deployed one-step C-PHAST predictor: observations to typed rollout

Require: Raw position history $\tilde{q}_{t-L_h+1:t}$, past input history $u_{t-L_h:t-1}$, current input u_t, deployment specification $\mathcal{S}_{\text{target}}$ with chart χ_{target}, shared parameters $\theta = (\theta_{\text{loc}}, \theta_C, \phi, \psi)$, time step Δt

Ensure: Predicted latent state $\hat{x}_{t+1}$ and decoded predicted position $\hat{q}_{t+1}^{\text{obs}}$

Normalize/wrap observations — deployment chart Eq. (3); topology and units declared by $\mathcal{S}_{\text{target}}$

$z_\tau \leftarrow \chi_{\text{target}}(\tilde{q}_\tau)$ for $\tau = t - L_h + 1, \ldots, t$

Lift model-coordinate history to latent pH state — q-only front end Eq. (21); observer Eq. (35)

$\hat{x}_t \leftarrow \Phi_{\phi,\psi}(z_{t-L_h+1:t}, u_{t-L_h:t-1})$ {re-run on the latest window with fixed weights}

Instantiate deployment composition — contract Table 6; port split Eq. (12)

$\hat{\mathbf{C}}^{(\mathcal{S}_{\text{target}})} \leftarrow \text{SkewCouple}_{\theta_C}(\mathcal{S}_{\text{target}})$ {Eq. (8)}

Decompose $\hat{x}_t$ into subsystem states $\{(q_i, p_i)\}_{i=1}^s$ according to the typed layout in $\mathcal{S}_{\text{target}}$

Stack local/control inputs as $\bar{\boldsymbol{u}}_t^{\text{loc}} \in \mathbb{R}^N$ and optional domain inputs as $\bar{\boldsymbol{u}}_t^{\text{dom}} \in \mathbb{R}^N$ using the port declarations in $\mathcal{S}_{\text{target}}$

Advance latent compositional dynamics — Algorithm 1; transition Eq. (22)

$\{(q_i^+, p_i^+)\}_{i=1}^s \leftarrow$ Algorithm 1 applied to $\{(q_i, p_i)\}_{i=1}^s$, $\bar{\boldsymbol{u}}_t^{\text{loc}}$, $\bar{\boldsymbol{u}}_t^{\text{dom}}$, and $\hat{\mathbf{C}}^{(\mathcal{S}_{\text{target}})}$

Restack the updated subsystem states as $\hat{x}_{t+1} = [(q_1^+, p_1^+); \ldots; (q_s^+, p_s^+)]$

Read out observed coordinates — projection and inverse chart in Eq. (22)

$\hat{z}_{t+1} \leftarrow \Pi_q(\hat{x}_{t+1})$

$\hat{q}_{t+1}^{\text{obs}} \leftarrow \chi_{\text{target}}^{-1}(\hat{z}_{t+1})$

return $(\hat{x}_{t+1}, \hat{q}_{t+1}^{\text{obs}})$

State refresh versus parameter adaptation. Algorithm 2 exposes the same fixed-weight state-refresh operation used by PHAST's online re-observation mechanism. In the takeover setting used for the main forecasting experiments, $\hat{x}_t$ is inferred once from a burn-in window and then rolled forward. When a fresh q-only history $\tilde{q}_{t-L_h+1:t}$ is available, the same coordinate map, observer, and canonicalizer can instead refresh the deployed latent phase state before a new rollout query, while keeping $(\theta_{\text{loc}}, \theta_C, \phi, \psi)$ fixed. Thus regime changes can appear as changes in inferred state, typed residuals, energy balance, or port-work diagnostics before any retraining is attempted. We do not interpret this as adaptive control; it is a reusable deployed-state update operator for q-only histories.

World-model physical critic interface. Once the deployed transition $G_\theta^{(\mathcal{S}_{\text{target}})}$ is available, a planner, controller, or protection layer can ask what would happen under one proposed next behavior. We use *candidate future* for that proposed behavior together with the rollout or consequence forecast it induces. The candidate may be a workspace trajectory, a torque sequence, a behavior primitive, a network switching action, a latent-action proposal, or a human-specified option. C-PHAST does not generate that candidate set. It supplies the structured query layer: if this option is executed under $\mathcal{S}_{\text{target}}$, what physical consequences would follow? A query therefore has four stages. A fresh observation window initializes or refreshes the deployed state, each candidate is rolled forward, typed consequences are evaluated, and planner-facing factors convert those consequences into objective-specific cards. Changing the objective changes the factor weights, hard envelopes, or priority rule, not the learned local dynamics. The closed-loop extension in which this rollout is repeatedly re-anchored by new measurements is evaluated directly in the cross-robot experiment: Table 8 reports the takeover and refreshed modes side by side, and Appendix I.5 traces the full curve between them. Figure 7 gives the conceptual pipeline; Appendix Tables 36, 38, and 39 list the real Spot data, input contract, and observable-to-factor mappings used by the hardware-data result.

With the pipeline in Figure 7 in view, the next equations write a candidate as a control sequence, which is the form used by the Isaac rollout and push-critic experiments. For logged Spot shadow mode, the same interface is queried with a behavior

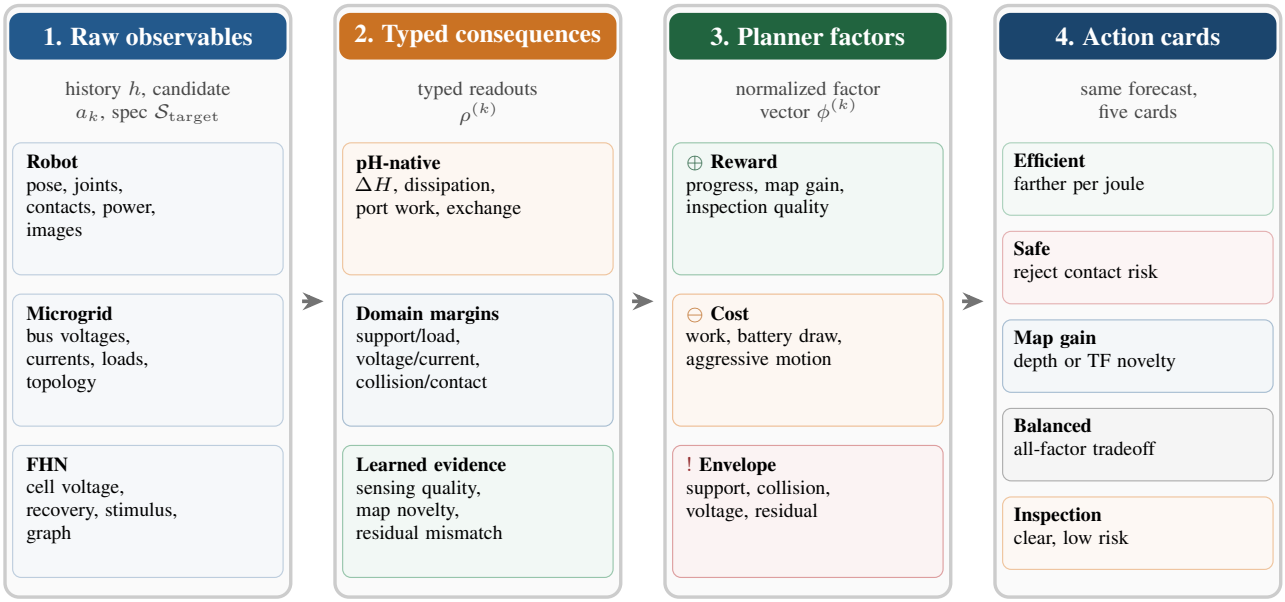

Figure 7. **From observables to action cards.** For one proposed next behavior a_k, such as fast traversal, slow scanning, support recovery, or arm inspection, C-PHAST first converts the current history and deployment specification into typed consequence readouts $\rho^{(k)}$, such as energy change, dissipation, port work, support or voltage margins, sensing value, and residual mismatch. Planner factors $\phi^{(k)}$ normalize these readouts into rewards, costs, barriers, or residual checks. An action card is the objective-specific view of that candidate future: efficient, safe, mapping, balanced, and inspection cards all reuse the same predicted physical consequences, then differ only in weights, vetoes, or priority rules. Appendix Tables 38 and 39 give the concrete Spot behavior-descriptor and observable-to-factor instantiation.

descriptor a_k instead of a full torque sequence. The descriptor summarizes an intended primitive such as fast traversal, slow scanning, arm inspection, or a conservative support-preserving motion; Appendix J.1 states the corresponding information contract. Given a finite candidate set of control sequences

$$U_t^{(k)} = \left(u_t^{(k)}, u_{t+1}^{(k)}, \dots, u_{t+H-1}^{(k)}\right), \qquad k = 1, \dots, K, \tag{23}$$

C-PHAST rolls each candidate forward from the same inferred latent state:

$$\hat{x}_t^{(k)} = \hat{x}_t, \qquad \hat{x}_{t+\ell+1}^{(k)} = G_\theta^{(\mathcal{S}_{\text{target}})}\left(\hat{x}_{t+\ell}^{(k)}, u_{t+\ell}^{(k)}\right), \quad \ell = 0, \dots, H-1. \tag{24}$$

The planner should not have to interpret arbitrary latent coordinates. For each candidate, C-PHAST first evaluates typed consequence readouts at their native physical scale,

$$\rho_{m,t+\ell}^{(k)} = \mathcal{O}_m\left(\hat{x}_{t+\ell}^{(k)}, u_{t+\ell-1}^{(k)}; \mathcal{S}_{\text{target}}\right), \qquad m = 1, \dots, M, \quad \ell = 1, \dots, H, \tag{25}$$

The index $\ell = 1$ refers to the first predicted state after applying $u_t^{(k)}$; in general, $\rho_{m,t+\ell}^{(k)}$ is evaluated at the state reached after action $u_{t+\ell-1}^{(k)}$. The channels can include predicted position, stored energy, dissipated power, port work, support/contact load, observation residual, interconnect exchange, and ordered damping-spectrum diagnostics. A planner factor is a declared scalar summary of one compatible readout family over the horizon,

$$\phi_a^{(k)} = \eta_a\left(h_a\left(\{\rho_{m,t+\ell}^{(k)} : m \in \mathcal{M}_a, \ \ell = 1, \dots, H\}\right) - b_a\right), \qquad a = 1, \dots, A, \tag{26}$$

Here $\mathcal{M}_a \subseteq \{1, \dots, M\}$ is the channel set used by factor a, and h_a reduces those selected readouts over the horizon to one physical quantity. For example, h_a may take a maximum safety violation over time, sum work over the horizon, evaluate a

terminal residual, or form an energy-balance mismatch from compatible channels. The boundary b_a is a target, budget, or envelope limit, and η_a converts the signed margin into the declared factor role: residual, cost, reward, barrier, or constraint slack. Thus the planner does not combine arbitrary latent coordinates. It combines typed physical factors whose sign, units, aggregation, and safety role are declared by the deployment specification.

Collecting these factors gives the planner vector $\phi^{(k)} = (\phi_1^{(k)}, \dots, \phi_A^{(k)})$. This vector is not a replacement for the candidate trajectory, map, goal, or controller state. It is the physical-consequence summary attached to candidate k, used by a downstream layer for ranking, filtering, or explanation. Hard envelopes are declared as functions $g_j : \mathbb{R}^A \to \mathbb{R}$, $j = 1, \dots, J$, with $g_j(\phi^{(k)}) \le 0$ feasible. The objective-conditioned action-card score is

$$s_k = w^\top \phi^{(k)}, \qquad k^\star = \arg \min_{k:\, g_j(\phi^{(k)}) \le 0 \ \forall j=1,\dots,J} s_k. \tag{27}$$

Here $w \in \mathbb{R}^A$ is the objective-specific factor-weight vector. For example, the factor set can include predicted envelope violations, energy or dissipation excursions, optional ordered damping-spectrum deviations, support/contact diagnostics exposed through declared domain ports, or port-power inconsistencies. These factors are typed physical readouts evaluated at their natural scale: stored or dissipated energy against energy budgets, port work against power flow, contact/support through declared ports, and modal damping through ordered spectra or eigenspaces rather than raw low-rank factors. The same predicted readout vector can support several planner modes: pure monitoring in shadow mode, scalar ranking over a finite candidate set, safety veto before ranking, or lexicographic selection such as "discard unsafe futures, then maximize task progress, then minimize energy." The recovery-candidate evaluation in Section 4.2 instantiates this abstract factor map with simple physical-envelope terms: predicted root-height margin, predicted maximum tilt, and predicted energy growth. If no candidate satisfies the hard envelopes, or if $s_{k^\star}$ exceeds a chosen soft threshold, the decision layer can reject the candidate set or fall back to a conservative action; otherwise it executes the first action $u_t^{(k^\star)}$ and repeats at the next timestep. This interface is generator-agnostic: candidates may come from a finite behavior library, a human operator, MPPI/CEM, MPC, or a learned policy. C-PHAST supplies the common query layer that rolls those futures forward and ranks their predicted physical consequences; Section 4.2 evaluates that layer with externally supplied recovery candidates.

Using the valid-window set $\mathcal{I}$ from the problem statement, each pair $(i,t) \in \mathcal{I}$ indexes trajectory i at a time t for which the history window $\tilde{q}^{(i)}_{t-L_h+1:t}$, the current input $u_t^{(i)}$, and the target $\tilde{q}^{(i)}_{t+1}$ are available. The base supervised term compares the decoded prediction against the observed target on the supervised channels declared by $\mathcal{O}_{\text{target}}$. The difference δ_q is induced by the deployment chart: for periodic coordinates we measure error with the wrapped S^1 difference, for positive coordinates the residual is evaluated through the declared positive-state chart, and for Euclidean coordinates $\delta_q(a,b) = a - b$:

$$\mathcal{L}_{\text{data}} = \frac{1}{|\mathcal{I}|} \sum_{(i,t)\in\mathcal{I}} \left\| \delta_q\left(\tilde{q}^{(i)}_{t+1}, \hat{q}^{\text{obs},(i)}_{t+1}\right) \right\|^2. \tag{28}$$

Equation (28) is the core supervised objective; the following terms are optional augmentations used only when the corresponding supervision or structure signal is available. For structured pH training we optionally augment it as

$$\mathcal{L} = \lambda_{\text{data}}\mathcal{L}_{\text{data}} + \lambda_{\text{energy}}\mathcal{L}_{\text{energy}} + \lambda_{\text{observer}}\mathcal{L}_{\text{observer}} + \lambda_{\text{pass}}\mathcal{L}_{\text{pass}} + \lambda_{\text{roll}}\mathcal{L}_{\text{roll}}, \tag{29}$$

The optional pH terms are evaluated on the predicted latent transition using start-of-step power estimates. For a sample (i,t), let $\hat{H}_t^{(i)} = H(\hat{x}_t^{(i)})$, $\Delta\hat{H}_t^{(i)} = \hat{H}_{t+1}^{(i)} - \hat{H}_t^{(i)}$, $\hat{v}_t^{(i)} = \partial H/\partial p(\hat{x}_t^{(i)})$, $\hat{y}_t^{\text{port},(i)} = \hat{v}_t^{(i)}$ for mechanical input ports, $\hat{P}_t^{\text{ext},(i)} = (u_t^{(i)})^\top \hat{y}_t^{\text{port},(i)}$, and $\hat{P}_t^{\text{diss},(i)} = -(\hat{v}_t^{(i)})^\top D(\Pi_q(\hat{x}_t^{(i)}))\hat{v}_t^{(i)} \le 0$, where Π_q extracts the model-coordinate configuration component used by the damping module. For typed C-PHAST deployments, these powers are computed per block or declared port and summed before entering the scalar loss. The passivity term is a one-sided hinge on surplus stored energy: it enforces the discrete inequality $\Delta\hat{H}_t \le \Delta t\, \hat{P}_t^{\text{ext}}$,

$$\mathcal{L}_{\text{pass}} = \frac{1}{|\mathcal{I}|} \sum_{(i,t)\in\mathcal{I}} \text{ReLU}\left(\Delta\hat{H}_t^{(i)} - \Delta t\, \hat{P}_t^{\text{ext},(i)}\right), \tag{30}$$

which reduces to $\text{ReLU}(\Delta\hat{H}_t)$ for closed ports. If an external port extracts power ($\hat{P}_t^{\text{ext}} < 0$), the same hinge penalizes predicted trajectories that fail to lose enough stored energy. The stricter energy-budget term then matches the discrete Hamiltonian rate to the full pH power balance, including modeled dissipation,

$$\mathcal{L}_{\text{energy}} = \frac{1}{|\mathcal{I}|} \sum_{(i,t)\in\mathcal{I}} \left| \frac{\Delta\hat{H}_t^{(i)}}{\Delta t} - \left(\hat{P}_t^{\text{diss},(i)} + \hat{P}_t^{\text{ext},(i)}\right) \right|. \tag{31}$$

This is a first-order training surrogate evaluated at start-of-step power; it is *not* the certified finite-step residual of Theorem B.1, whose $\mathcal{O}(\Delta t^3)$ ledger uses trapezoidal boundary accounting, and the two agree only to first order in Δt. When simulator velocities are available, the q-only observer can also be supervised directly,

$$\mathcal{L}_{\text{observer}} = \frac{1}{|\mathcal{I}|} \sum_{(i,t)\in\mathcal{I}} \left\| \hat{\dot{q}}_t^{(i)} - \dot{q}_t^{(i)} \right\|^2 . \tag{32}$$

Finally, the optional rollout term unrolls the model from the same burn-in state for horizons $\mathcal{H}_{\text{train}}$:

$$\mathcal{L}_{\text{roll}} = \frac{1}{|\mathcal{I}|\,|\mathcal{H}_{\text{train}}|} \sum_{(i,t)\in\mathcal{I}} \sum_{h\in\mathcal{H}_{\text{train}}} \left\| \delta_q\left(\tilde{q}_{t+h}^{(i)}, \hat{q}_{t+h|t}^{\text{obs},(i)} \right) \right\|^2 . \tag{33}$$

Matched-protocol comparisons use only $\mathcal{L}_{\text{data}}$ unless stated otherwise, while the full PHAST-family recipe may additionally activate the energy and observer terms when the corresponding labels or pH diagnostics are available; Appendix F records benchmark-specific settings and ablation-only variants.

Observer for velocity estimation. We infer velocities from position history using a causal observer. Following the state pipeline, after the deployment chart is applied we write $q_\tau = z_\tau$ for the generalized coordinate used by the local mechanical pH block. For the window ending at t, write $a = t - L_h + 1$. We first compute finite-difference velocities over that window:

$$\dot{q}_a^{\text{fd}} = 0, \qquad \dot{q}_\tau^{\text{fd}} = \frac{\Delta(q_\tau, q_{\tau-1})}{\Delta t}, \quad \tau = a+1, \dots, t, \tag{34}$$

where $\Delta(\cdot,\cdot)$ is the coordinate-aware difference induced by χ_{target}: standard subtraction for Euclidean charts and the wrapped difference for periodic charts. A learned residual corrects finite-difference errors:

$$\hat{\dot{q}}_{a:t} = \dot{q}_{a:t}^{\text{fd}} + o_\phi\big(q_{a:t}, \dot{q}_{a:t}^{\text{fd}}\big) , \tag{35}$$

where o_ϕ is a temporal convolutional network (TCN) that outputs a per-timestep correction; its value at index τ is causal in the prefix ending at τ. The observer is *shared per subsystem*: the same TCN weights process each joint's (or leg's) position history independently, making the observer compositional and compatible with variable N.

Canonicalization and rollout. The canonicalizer C_ψ maps $(q, \hat{\dot{q}}) \mapsto (q, \hat{p})$. In practice we use either a diagonal-mass canonicalizer, $\hat{p} = M\hat{\dot{q}}$, when an effective mass model is specified, or the identity map $\hat{p} = \hat{\dot{q}}$ when no reliable mass model is available. This is the lift from model-coordinate evidence to the latent pH state $\hat{x}_t = (q_t, \hat{p}_t)$ that the dynamics can advance. Given the inferred initial state at the end of burn-in, write that end time as t_0. We roll out open-loop in the latent state while applying the known input sequence:

$$\hat{x}_{t_0} = C_\psi(q_{t_0}, \hat{\dot{q}}_{t_0}), \quad \hat{x}_{\tau+1} = G_\theta^{(\mathcal{S}_{\text{target}})}(\hat{x}_\tau, u_\tau), \quad \hat{q}_\tau^{\text{obs}} = \chi_{\text{target}}^{-1}(\Pi_q(\hat{x}_\tau)) , \quad \tau \geq t_0, \tag{36}$$

where $\Pi_q(q,p) = q$ extracts model-coordinate positions and $\hat{q}_\tau^{\text{obs}}$ is the decoded observation-space prediction.

At inference, the primary mode is *takeover*: infer initial state from a burn-in window, then integrate open-loop. This q-only interface is the relevant transfer setting in robotics: when deploying to a new morphology, we can usually reuse encoders and commanded inputs, but we cannot assume calibrated velocity sensors on the target system. The observer generalizes across morphologies because it operates per-joint in canonical coordinates.

3.7. Transfer Setting and Assumptions

The method covers two dynamics-redeployment patterns studied in the paper: *homogeneous repeated-block transfer*, where one local block type is instantiated many times (planar arms, repeated leg/block decompositions, and FitzHugh–Nagumo excitable cells), and *heterogeneous typed-block transfer*, where the system is assembled from a fixed finite library of block types (microgrid DGU and line blocks). These assumptions scope the deployment contract for the dynamics model: they state what may change at transfer, what must remain fixed within a block type, and where the continuous-time pH idealization becomes an approximation under finite-step observed-coordinate rollouts. The Spot-with-arm result uses the same typed consequence interface for real robot telemetry and action cards, but it is not a subsystem-count or topology-transfer benchmark.

1. **A1. Finite typed block library.** Each system instance is composed from either one repeated block type or a fixed finite library $\mathcal{T}$ of typed local blocks. Parameter sharing is within type, not across unrelated types.

2. **A2. Within-type local dimension is fixed.** Every block type $\alpha \in \mathcal{T}$ has a fixed local state dimension d_α. Transfer may change components of the deployment specification $\mathcal{S}_{\text{target}}$, such as subsystem count, typed layout, topology, observation interface, or instance metadata, but it does not change the internal dimension of an existing type.

3. **A3. Transfer respects the typed decomposition.** In repeated-block settings, transfer resizes a family of same-type blocks, as in planar arms, homologous leg-count benchmarks, and FHN cell-count transfer. In fixed-layout embodiment transfer, the typed layout is reused while instance metadata changes, as in ANYmal-D to Go2. In heterogeneous settings, transfer reuses the same type library while changing only declared components of $\mathcal{S}_{\text{target}}$, such as typed layout, topology, observation interface, or instance metadata.

4. **A4. Mechanical rollout state restriction.** In the arm, MuJoCo, and Isaac Lab dynamics benchmarks, the learned pH rollout state contains actuated coordinates only; floating-base pose is excluded in the legged experiments. The Spot-with-arm result uses the typed consequence interface on real telemetry rather than this q-only compositional rollout state.

5. **A5. Approximate separability.** The per-subsystem decomposition is exact when the Hamiltonian is separable after canonicalization ($H = \sum_i H_i$); for systems with strongly configuration-dependent cross-coupling mass, it is an approximation whose quality we verify empirically.

6. **A6. Continuous-time vs. deployed rollout gap.** Power-preserving interconnection is exact in continuous time with exact port variables; under finite-step observed-coordinate rollouts and learned canonicalization, we treat it as a structural prior and report coupling-power diagnostics (Table 25).

These assumptions complete the transfer contract for the dynamics model.

3.8. Deployment-Contract Guarantees

The contract carries three levels of guarantee—per step, over a horizon, and at the decision—each tied to a quantity the implementation computes and validated against it (formal hypotheses and proofs are in Appendices B, C, and D; reproduction scripts accompany the paper). Write $\Psi_{\Delta t}$ for the deployed one-step map, $\varphi_{\Delta t}$ for the true flow, and e_k for the aligned deployed-vs-true state error.

Inherited structure (preserved, not new). The skew interconnection injects zero internal power (Theorem 3.1); this classical port-Hamiltonian composition guarantee is *preserved* by C-PHAST's typed neural, deployment-time instantiation, not introduced by it.

Per-step energy-balance consistency. One structure-preserving step respects the energy balance to third order, $H(x_{t+1}) - H(x_t) = W_t^{\text{port}} - D_t + \mathcal{O}(\Delta t^3)$, with an additive first-order observer/chart term under q-only sensing (Theorem B.1). The model logs the rearranged residual $H(x_{t+1}) - H(x_t) - (W_t^{\text{port}} - D_t)$ (its ΔH + dissipation − port work channel), which the theorem identifies as the $\mathcal{O}(\Delta t^3)$ defect; that scaling is measured against the analytic flow (Table 17). This is a *model-internal* consistency result—for the model's own H, R, B, not a claim that the learned energy equals physical stored energy—parallel to, not a component of, the transfer bound below; it yields a genuine discrete passivity inequality only on steps where dissipation exceeds the defect (Appendix B).

Horizon transfer. Over h steps the deployed rollout tracks the true trajectory as

$$e_h \;\le\; q^h\,\epsilon_{\text{lift}} + S_h(q)\,\bar{\delta}, \qquad S_h(q) = \tfrac{1-q^h}{1-q}\;\; (q<1), \quad h\;\; (q=1), \tag{37}$$

where ϵ_{lift} is the observer lift error and $\bar{\delta}$ the per-step model defect—local error plus a coupling term $\|K(\mathcal{S})\|\,\delta_{\text{edge}}$ through a generic interconnection operator $K(\mathcal{S})$ (specializing to the incidence norm $\|B_{\text{inc}}(\mathcal{S})\|$, first power, in the microgrid) plus an $\mathcal{O}(\Delta t^3)$ *state*-truncation term (the deployed one-step map versus the true flow; distinct from the per-step *energy* defect above, Theorem C.2). Crucially, the contraction factor q is *not* a consequence of passivity or symplecticity (a symplectic step preserves a two-form, not a positive-definite norm; the limiting value $q = 1$ is exhibited by the specific conservative

Table 7. Guarantee ledger. Each claim carries its *earned* status: architectural results are exact by construction; the finite-step, transfer, and decision guarantees are formally certified only for the analytic KNOWN-regime instances, and are empirically calibrated—not universal—for trained and hybrid deployments.

Claim	Result	Required assumptions	KNOWN cases	Trained cases
Internal power cancellation	Thm 3.1	exact ports, skew coupling	exact	architectural
Finite-step energy balance	Thm B.1	smooth fixed mode, 2nd-order step	verified/certified	diagnostic
Horizon tube	Thm C.2/C.4	regional contraction, model defect	selected certificates	empirically calibrated
Decision margin	Thm D.2	valid state/readout/map bounds	conditional certificate	empirical margin
Energy–Casimir stability	Thm A.1/A.4	exact port, local basin	theorem	q-only compatibility test

linear oscillator, not implied for every passive conservative system—Remark C.6), but is the one-step contraction verified in a Lyapunov metric on the deployment's stable operating region—interval-certified for the linear deployments and measured for the nonlinear pendulum. When $q < 1$ (damped, in-basin) the bound is uniform in the horizon. Across the tested bounded-degree deployments ($N \in \{3, 4, 6\}$) the verified contraction factors remained below one in a per-block RMS metric; we make no asymptotic graph-size-uniform claim, and this bounded-count behavior is the transfer property the count/topology experiments exhibit.

Decision reliability. Each typed consequence inherits the state bound through its readout: $|\hat{\rho}_m - \rho_m| \le \Delta_m$, with Δ_m a certified readout-Lipschitz constant times e_k plus a measured learned-vs-physical map-error term (ϵ^{map}=0 for an exact analytic readout and nonzero for a learned one; non-smooth contact/switching channels are treated mode-conditionally). A linear planner score then obeys $|\hat{J}_w - J_w| \le \Delta_J$, so candidate *rankings* are preserved when the predicted score gap exceeds Δ_J, the predicted optimum has regret at most $2\Delta_J$, and a hard constraint is certified by margin tightening, $\hat{g}_j \le -\Delta_{g_j} \Rightarrow g_j \le 0$ (Proposition D.1, Theorem D.2). These are deterministic decision guarantees *whenever* the state-tube, readout, and map-error constants are valid upper bounds on the operating region: certified for the analytic KNOWN-regime channels, and empirically calibrated for trained or hybrid deployments—for which ϵ^{map} is not certified here. This is what makes the typed-consequence interface of Figure 1 a decision object with *conditional* guarantees, not merely a forecast.

Table 7 states each guarantee with the status it actually carries—exact by construction, certified on the KNOWN cases, or empirically calibrated on the trained ones—so that architectural exactness, interval certification, and empirical calibration are never conflated.

Section 5 studies a separate controller-compatibility consequence of the same pH port interface. We next evaluate whether shared local blocks, typed interconnection, and the deployed observation-to-rollout interface translate into empirical generalization across new system instances and into typed consequences useful for downstream decisions.

4. Experiments and Results

We evaluate C-PHAST by changing the deployed system while keeping the learned local port-Hamiltonian blocks fixed. The guiding question is simple: when a new instance arrives, can the model reuse the same typed local physics, rebuild only the declared composition, and still produce rollouts or physical factors that a decision layer can use?

The experiments form a deployment-stress ladder rather than a list of independent benchmark demos. **Isaac Lab** is the lead robotics rollout result: ANYmal-D and Go2 share a 4×3 leg layout, but differ in mass, geometry, inertia, and contact response. The Go2 push critic then asks whether the same rollout object can support decision-time factors as well as prediction error. **Real Spot-with-arm logs** provide the hardware result: C-PHAST forecasts physically named future consequences from real robot telemetry and turns them into objective-conditioned action cards. **Planar arms** and **MuJoCo** test repeated mechanical block count changes. **FitzHugh–Nagumo** tests the same repeated-block recomposition outside robotics on an excitable circuit family. **DC microgrids** are the boundary case: heterogeneous DGU/line blocks must remain useful when the graph or sensing interface changes, even though the full-state known-graph regime remains stronger for N-PHDAE.

This ordering also explains the metric choice. One-step prediction error is useful, but it is too narrow for a model meant to feed planners, protection logic, and design queries. We report one-step MSE, but treat multi-step rollout behavior as the more informative signal when the two disagree. In robotics this asks whether predictions remain useful on a new body; in electrical systems it asks whether the model remains useful for regulation, monitoring, or design evaluation when topology

or sensing changes. Throughout, "rollout stability" is empirical rather than theorem-level: the model does not diverge under the tested rollout protocol, even though all tested models remain over-smoothed on contact dynamics (Appendix I.1).

We use two recurrent transfer labels before introducing the microgrid-specific regimes. *Count transfer* resizes a repeated subsystem family while preserving local semantics and local block dimension: joint modules, homologous leg blocks, or repeated excitable-cell blocks. *Cross-robot transfer* keeps the typed layout fixed but changes the physical instance metadata Ξ_{target}. The robotics benchmarks therefore test the learned-dynamics substrate that contact planners and controllers would query. The microgrid subsection adds topology reconfiguration and voltage-only sensing, because that domain changes by graph and observation interface in addition to count.

4.1. Experimental Setup

This setup subsection makes the transfer questions concrete before the result tables arrive. We first define the robotics and mechanics settings where the same learned local blocks are reused across a new body or a new part count. The Spot, FHN, and microgrid subsections then introduce their own domain-specific contracts where the evidence needs additional data or graph context.

Isaac Lab benchmark (primary). The lead robotics test changes the body without changing the declared leg layout. **ANYmal-D** (ANYbotics, $\approx$50 kg) and **Unitree Go2** ($\approx$15 kg) both have 4 legs $\times$ 3 actuated joints per leg: hip abduction/adduction (HAA), hip flexion/extension (HFE), and knee flexion/extension (KFE). Their masses, link lengths, inertia tensors, and contact responses differ substantially. This makes ANYmal-D$\rightarrow$Go2 a fixed-layout embodiment transfer: the typed leg blocks and observation/action layout are reused, while the physical instance metadata and response change.

This fixed-layout distinction matches Isaac Lab's own sim-to-real guidance for fixed-layout manipulation. When the task topology is fixed, the difficult shift is system response, so the recommended randomization targets actuator gains, joint friction / stiction, and initial or object poses rather than a new graph of parts (NVIDIA Isaac Lab Developers, 2026). Our robotics PARTIAL regime follows that information contract: it uses the grouped articulation, q-only observation/action interface, and coarse robot metadata exposed by current asset stacks, without assuming full contact- and actuator-level response is known a priori. We therefore treat Isaac Lab as the main embodiment-mismatch result; the planar-arm "stale URDF" sweep in Appendix G.10 is a controlled mass-prior corruption sanity check.

For data collection, we train one RSL-RL locomotion policy per robot (Rudin et al., 2022) (PPO, 1500 iterations, 4096 parallel envs). We then collect 500 trajectories per robot of length T=200 at Δt=0.02 s with mild action noise (σ=0.05). Models are trained on ANYmal-D (350 train, 50 val) and tested *without retraining* on Go2 (100 test trajectories).

MuJoCo legged benchmark. The MuJoCo family removes the fixed-body complication and asks a cleaner count question. We use a biped (N=2 legs), quadruped (N=4), and hexapod (N=6), each built from the same 2-DOF leg template. Only the number of repeated leg blocks changes, so this benchmark tests whether a learned leg block and coupling rule can be resized without changing parameters. Trajectories are generated by applying i.i.d. random torques clipped to $[-1, 1]$ at Δt=0.02 s for T=250 steps. Models train on the quadruped and test without retraining on the biped and hexapod.

Robot arm benchmark. The robot arm is the controlled mechanical test case for count transfer and power-preserving interconnection. It is a planar N-joint arm with diagonal inertia, viscous damping, separable gravity, q-only observations, sinusoidal torque inputs, and a fixed skew-symmetric inter-joint coupling in the simulator. That hidden coupling creates a useful diagnostic: it can redistribute effort across joints, but because it is skew-symmetric it should not inject net energy. Appendix G gives the exact simulator contract and Figure 16 provides the schematic. Train on 200 trajectories with N=4; test on 50 trajectories for each $N \in \{2, 3, 4, 5, 6, 8\}$.

Models compared. The baselines are chosen to answer three alternative explanations. First, compositional baselines (**Shared-GRU** (Cho et al., 2014), **MPNN** (Gilmer et al., 2017), **DeepSets** (Zaheer et al., 2017)) ask whether local weight sharing alone explains transfer. Second, non-compositional baselines in Isaac Lab ask whether port-Hamiltonian structure helps even when the typed layout is fixed: **Monolithic PHAST** (Bhardwaj & Bajaj, 2026) (18,473 params), **Fixed-dim GRU** (Cho et al., 2014) (53,068 params), **Neural ODE** (Chen et al., 2018) (24,344 params), and a parameter-free **persist** baseline ($\hat{q}_{t+1} = q_t$). Third, robot-arm and MuJoCo ablations vary the coupling rule itself: **neural coupling**, **no learned coupling**, and **power-preserving skew coupling**, alongside fixed-dimension sequence baselines (S5 (Smith et al., 2023), LRU (Orvieto et al., 2023), LinOSS (Hasani et al., 2024), GRU (Cho et al., 2014), PHAST (Bhardwaj & Bajaj, 2026)

regimes). For robot arm and MuJoCo, all compositional models are trained on $N{=}4$ and evaluated at all N; FHN and microgrid use $N{=}3$ source systems as stated in their benchmark paragraphs below.

Metrics. The metrics separate local fit, rollout behavior, and structure diagnostics. For each benchmark, prediction error is measured in the physical coordinate or readout vector reported by that table; write this evaluation vector as $\xi_t \in \mathbb{R}^{D_\xi}$. This is only a metric symbol: the method notation remains x_t for the latent pH state and z_t for charted observations. One-step MSE measures immediate prediction,

$$\mathrm{MSE} = \frac{1}{(T-1)D_\xi} \sum_t \|\hat{\xi}_{t+1} - \xi_{t+1}\|^2.$$

Takeover rollout uses a 5-step burn-in context and then integrates open-loop for H steps. The refreshed rollout re-anchors the same model in observations every K steps using the state-refresh operator of Section 3; unless stated otherwise we report $K{=}5$, one refresh per $0.1\,\mathrm{s}$ at the $50\,\mathrm{Hz}$ control rate. For skew-symmetry checks, the coupling power residual is $\mathbb{E}[|(y^{\mathrm{port}})^\top \hat{u}|]$. We define divergence as $\mathrm{MSE}^{\mathrm{take}}(H) > 10^3$; diverged seeds are excluded from mean$\pm$std and the fraction noted. Whenever instability is central to the comparison, tables report it explicitly through *diverges* entries or footnotes.

Training. Every row of the Isaac transfer table uses the same *matched training protocol*: one data split, one AdamW schedule, 50 epochs, and a one-step MSE objective, so that differences in transfer reflect architecture rather than supervision. In this protocol, pH models do not receive energy-budget or observer-velocity auxiliary losses. The full structured recipe (200 epochs, $\lambda_{\text{energy}}{=}1.0$, $\lambda_{\text{observer}}{=}1.0$) is reported separately in Appendix I.5, because its effect turns out to depend on the rollout mode. The Isaac artifacts in this section are regenerated end to end: the original trajectory caches and checkpoints were not retained, so both robots were re-collected with the stated pipeline (locomotion policies trained for 1,500 PPO iterations at 4,096 environments, then 500 trajectories of 200 steps per robot) and every row was retrained from a committed configuration. The regenerated persist levels (0.045 on ANYmal-D, 0.152 on Go2) closely match the originally reported regime, and each number in the table can be regenerated from a single script and its results manifest.[1] Elsewhere we use the benchmark-default recipe stated here unless a caption explicitly notes a rerun. Seeds: robot-arm 10, MuJoCo 3, FHN 3, Isaac Lab 5 for the main transfer table (see individual table captions for other Isaac analyses).

4.2. Cross-Robot Transfer (Isaac Lab)

Deployment stress (Fig. 3): LOCAL LAW LEARNED (embodiment / response shift), under Q-ONLY SENSING. The Isaac Lab result is the cleanest embodiment-mismatch test in the paper. A model sees ANYmal-D trajectories during training, then must roll out Go2 without target retraining. Because both robots share the same 4×3 typed leg layout, the input/output shape is not the hard part; the hard part is whether the learned flow remains useful when the mass distribution, geometry, inertia, and contact response change. Every model in Table 8 is trained under the matched protocol defined above and then evaluated in both rollout modes: takeover with a five-step burn-in, and the refreshed rollout that re-anchors in measurements every five steps.

Read Table 8 by rollout mode first and by model family second. The takeover columns are the open-loop stress test: after the burn-in, the model runs on its own predictions for fifty steps. This is the regime a physical critic inherits when it scores candidate futures, since a hypothetical future has no measurements to refresh from. The refreshed columns re-anchor the same rollout in observations every five steps, which is the regime a deployed monitor inherits under the state-refresh operator of Section 3. Within each mode, the top block compares compositional models and the bottom block adds fixed-layout baselines, so we can ask whether port-Hamiltonian structure helps even when no resizing is required. Figure 8 then checks whether the takeover numbers reflect stable rollout behavior or late-horizon collapse.

Takeover transfer. Under the matched protocol, C-PHAST's one-step MSE degrades by $2.4\times$ from ANYmal-D to Go2 ($7.1 \to 17.3 \times 10^{-4}$), while its takeover rollout error degrades by only $1.6\times$ ($0.045 \to 0.073$). The horizon plot gives the qualitative reason: C-PHAST stays nearly flat on both robots, whereas the recurrent and graph baselines accumulate error quickly once they run on their own predictions. That contrast is not explained by source fit. The recurrent baselines are the stronger source-robot forecasters, with the fixed-dimension GRU reaching the lowest ANYmal-D rollout error (0.005) and Shared-GRU close behind (0.014); they then degrade $21\times$ and $13\times$ on Go2. The per-leg MPNN matches C-PHAST's

[1] All Isaac rows share the same train/validation/test split and optimizer family. Robot arm hyperparameters are in Appendix G.5; Isaac Lab and MuJoCo settings are detailed in Appendix H.

Table 8. Cross-robot transfer on Isaac Lab (PhysX), canonical regeneration. Train on ANYmal-D and test on Go2 without retraining; all rows use the matched protocol (5 seeds, 50 epochs, one-step MSE only) and are evaluated in both rollout modes. Takeover: five-step burn-in, then open loop for $H{=}50$. Refreshed: the same rollout re-anchored in measurements every $K{=}5$ steps. Rollout MSE in rad^2; best per column in **bold**. A superscript on a model name gives the fraction of seeds whose training diverged ($\mathrm{MSE} > 10^3$); those seeds are excluded from takeover means and noted, while refreshed means include all seeds, which is why the Shared-GRU refresh cells carry a large deviation. The full structured recipe for C-PHAST is reported in Appendix I.5.

Model	Params	Takeover rollout ($H{=}50$)			Refreshed rollout ($K{=}5$)		
		ANYmal-D	Go2	gap	ANYmal-D	Go2	gap
Compositional (per-leg):							
C-PHAST	40,034	0.045 ± 0.000	$\mathbf{0.073 \pm 0.001}$	1.6×	0.0066 ± 0.000	$\mathbf{0.015 \pm 0.000}$	2.3×
Shared-GRU $^{1/5}$	60,804	0.014 ± 0.003	0.178 ± 0.025	13×	0.048 ± 0.087	0.047 ± 0.048	1.0×
MPNN (per-leg)	42,692	0.638 ± 0.686	263 ± 361	—	0.0069 ± 0.000	0.022 ± 0.001	3.3×
DeepSets $^{1/5}$	26,116	0.071 ± 0.050	28.1 ± 45.7	—	0.0083 ± 0.000	0.021 ± 0.001	2.6×
Non-compositional (fixed 12-DOF input):							
Monolithic PHAST	34,601	0.039 ± 0.002	0.145 ± 0.016	3.7×	$\mathbf{0.0035 \pm 0.000}$	0.022 ± 0.000	6.4×
Fixed-dim GRU	43,020	$\mathbf{0.005 \pm 0.000}$	0.112 ± 0.005	21×	$\mathbf{0.0035 \pm 0.000}$	0.082 ± 0.003	23×
Neural ODE $^{1/5,\ 3/5}$	14,860	357 ± 391	18.9 ± 13.9	—	0.080 ± 0.016	1.04 ± 0.35	13×
Reference (no learned parameters):							
Persist ($\hat{q}_{t+1}{=}q_t$)	0	0.045	0.152	3.4×	0.049	0.136	2.8×

compositional granularity but collapses on Go2, DeepSets loses a diverged seed and still lands far above persist, and one Shared-GRU seed diverges during training itself. Thus the primary takeover result is not that C-PHAST fits the source robot best; it is that the pH-structured rollout remains usable on the new body.

Two rollout modes, two deployment roles. The refreshed columns complete the picture. Once the rollout is re-anchored in measurements every five steps, every family that survived training improves, and C-PHAST attains the lowest absolute Go2 error (0.015, against 0.021 to 0.082 for the alternatives). The two modes answer different deployment questions. A candidate future cannot be refreshed, because its measurements do not exist yet, so takeover stability is the property that the physical-critic interface of Section 3.6 inherits. A monitored system is refreshed continuously, so the refreshed columns measure what state tracking inherits. C-PHAST is the only family competitive in both modes, and its two ingredients act in different places: the typed per-leg decomposition improves refreshed transfer over its monolithic counterpart (0.015 versus 0.022 on Go2, with a 2.3× rather than 6.4× gap), while the pH structure is what keeps the open-loop rollout bounded, since the typed baselines without it collapse in takeover. Appendix I.5 traces the full curve between the two modes and shows how takeover skill depends on the information in the burn-in window.

Disentangling pH structure from compositionality. Because the typed layout is fixed in this benchmark, the bottom block of Table 8 asks a sharper question: does pH structure help even without compositional resizing? The pattern is clear. Monolithic PHAST transfers at 3.7× takeover degradation, while the fixed-dimension GRU degrades 21×. Within the compositional family, C-PHAST transfers at 1.6× while Shared-GRU degrades 13× and MPNN and DeepSets collapse. A parameter-free persist baseline ($\hat{q}_{t+1}{=}q_t$) contextualizes absolute error: its takeover rollout MSE of 0.045 on ANYmal-D and 0.152 on Go2 is the level at or above which a model provides no useful dynamical skill beyond holding the last observation. These persist levels are a property of the behavior policies as much as of the robots, since they measure how vigorously each policy moves within the evaluation window; an under-trained target policy inverts their ordering, which is why the protocol pins the policy training budget together with the seeds. The Neural ODE diverges outright on the majority of Go2 seeds, so continuous-time parameterization alone is not enough. Parameter counts are not matched across cells (14,860–60,804), so this is an inductive-bias comparison rather than a capacity-controlled theorem.

Accuracy–transfer tradeoff. At this point it is tempting to rank models by source-robot accuracy. That would choose the fixed-dimension GRU: it achieves a 9× lower in-distribution takeover error on ANYmal-D (0.005 vs. 0.045). The Go2 column reverses that choice because the source fit does not survive the robot change. This pattern is consistent with a model that captures source-specific contact response rather than a transferable physical flow. C-PHAST has its own hard case: Isaac Lab locomotion is hybrid, with stance/swing mode switches and impulsive ground-reaction forces that are not represented by a smooth Hamiltonian $H(q, p)$ (Brogliato, 1999). A systematic observer ablation (Appendix I.2) shows that improving

source-side contact fit does not remove the transfer tradeoff. Section 4.4 asks the natural follow-up: if Shared-GRU is given target Go2 trajectories, does it catch up?

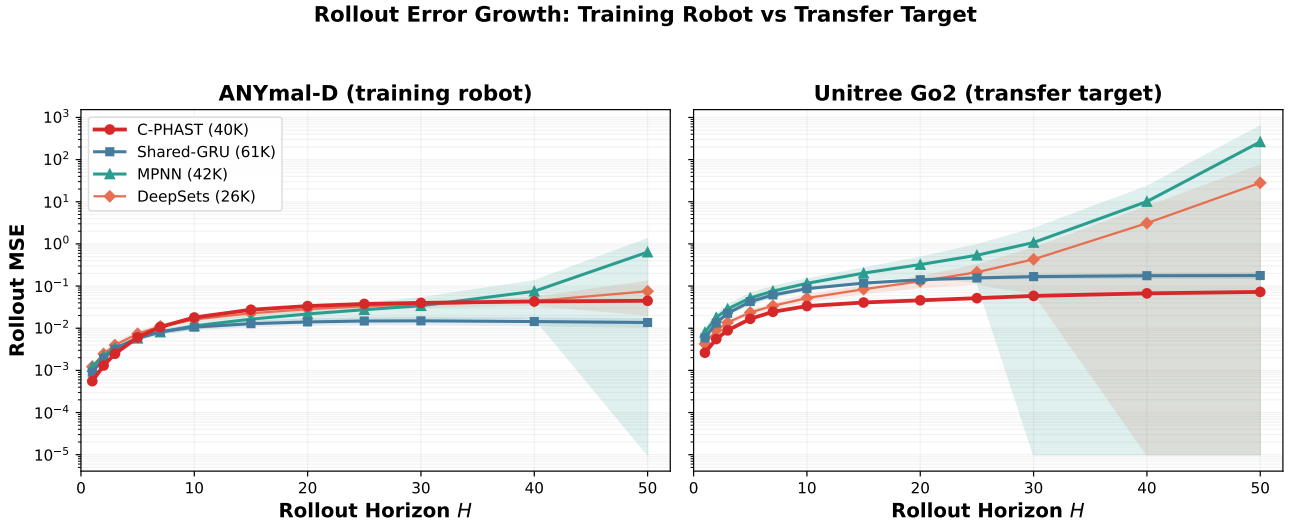

Figure 8. Takeover rollout MSE vs. horizon H on ANYmal-D (left) and Go2 (right), canonical regeneration. C-PHAST remains nearly flat on both robots, while the recurrent and graph baselines accumulate error rapidly on Go2. Shaded bands: $\pm 1\sigma$ over 5 seeds; training-diverged seeds excluded.

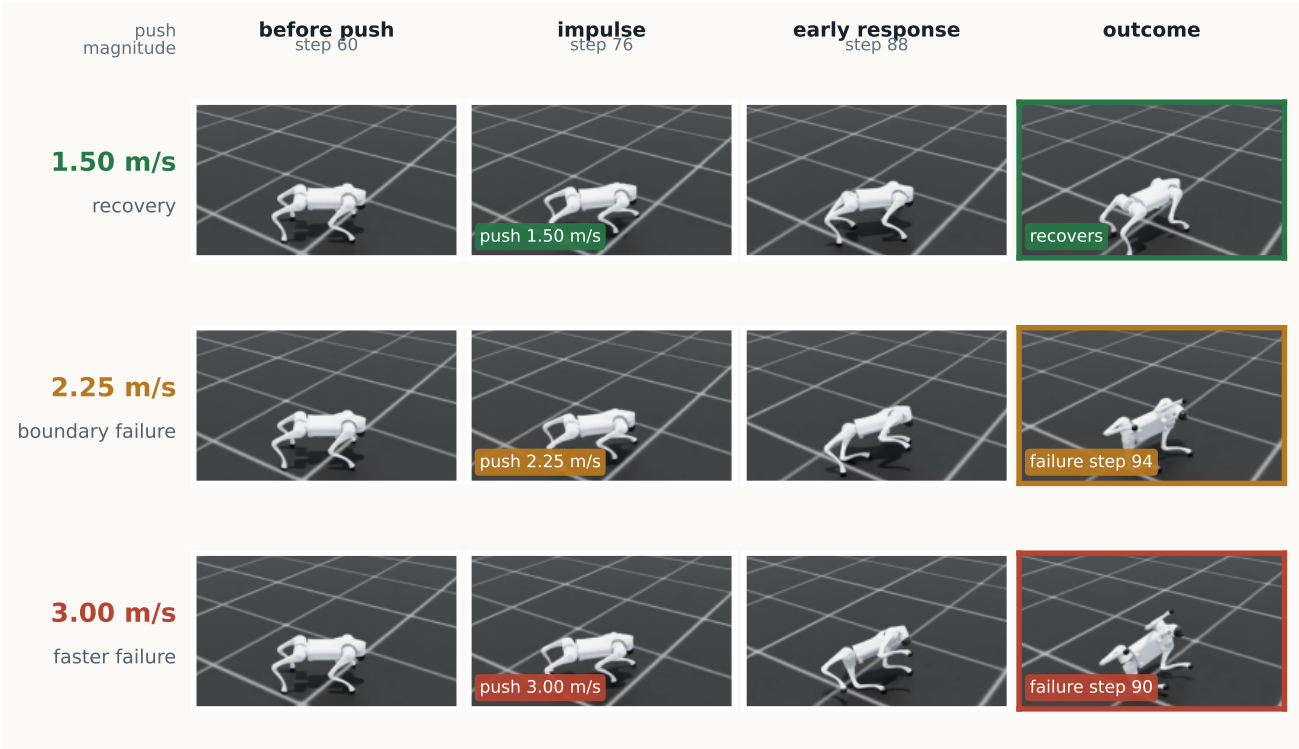

Figure 9. **Closed-loop Go2 recovery boundary under lateral pushes.** The same policy-controlled Go2 recovers from a $1.50\,\mathrm{m/s}$ lateral impulse, reaches a boundary-failure regime at $2.25\,\mathrm{m/s}$, and fails sooner at $3.00\,\mathrm{m/s}$. This perturbation envelope provides the decision-loop stress test for the C-PHAST critic in Table 9: the critic emits unsafe-future signals before simulator-level failure on the failing runs.

Model-predictive physical critic. *Deployment stress (Fig. 3):* the off-axis DECISION-FACING SCORING capability. The previous result asks whether the rollout stays useful on a new robot body. The critic test asks whether that same rollout object can be queried before choosing an action. We instantiate the critic interface of Section 3.6 in Isaac Lab under lateral-push perturbations. The deployment scene is a policy-controlled Go2 that receives a lateral impulse and must either recover or fall. Figure 9 visualizes the $+\hat{y}$ envelope: the same policy recovers at $1.50\,\mathrm{m/s}$, fails near the recovery boundary at $2.25\,\mathrm{m/s}$, and fails faster at $3.00\,\mathrm{m/s}$.

The decision loop is candidate-conditioned. At each step, an external layer supplies a small set of control sequences: the policy continuation and several alternatives. C-PHAST rolls each future forward from the same state, evaluates typed physical consequences, and emits two logged events: an *unsafe-future signal*, when the policy continuation is predicted to violate the physical envelope, and an *action-change event*, when the decision layer executes a lower-risk supplied candidate.

Table 9 extends the push test to four directions and five magnitudes. Across the 20 closed-loop Isaac Lab runs, 14 terminate in simulator-level failure and C-PHAST emits an unsafe-future signal before all 14 failures, with $7/15.6/29$ control steps of min/mean/max lead. There are also two false warnings among the six recoverable pushes, so this result is strongest as an early physical filter and typed branch scorer.

Table 9. **Directional physical-critic sweep under Go2 lateral pushes.** We execute the C-PHAST critic in the Isaac Lab action loop for four push directions and five magnitudes. *Failure boundary* is the smallest tested magnitude in that direction that produces simulator-level failure. *Unsafe futures* counts failing runs for which C-PHAST predicts the policy continuation is unsafe before failure. *False warnings* counts recoverable runs with an unsafe-future signal. *Lead steps* reports min/mean/max lead between first unsafe-future signal and simulator-level failure. *Action changes* counts failing runs where the decision layer executed a lower-risk supplied candidate.

Push dir.	Failure boundary	Unsafe futures	False warnings	Lead steps	Action changes
$+x$	≥ 1.75 m/s	$3/3$	$1/2$	$14/15.3/16$	$1/3$
$-x$	≥ 1.50 m/s	$4/4$	$1/1$	$7/12.8/22$	$1/4$
$+y$	≥ 1.75 m/s	$3/3$	$0/2$	$11/15.3/18$	$2/3$
$-y$	≥ 1.50 m/s	$4/4$	$0/1$	$13/19.0/29$	$3/4$
All	–	$\mathbf{14/14}$	$2/6$	$7/15.6/29$	$7/14$

The table gives the alarm view; the branch view is a same-state comparison among supplied futures. Because the decision layer can only choose among futures it is given, C-PHAST's role is to put physical labels on those futures before selection. The resulting interface can run inside a decision loop, warn before failure, and rank supplied futures by typed physical consequences. Figure 10 shows one same-state branch from this interface. A richer recovery library, formal safety filter, or MPC planner would consume the same typed scores. Appendix J.2 decomposes the warning into typed residual channels and compares it with scalar threshold and learned measured-state alarms. Appendix J.2.1 reports a narrower damping-spectrum diagnostic that behaves as a high-confidence typed channel rather than a replacement for the full critic.

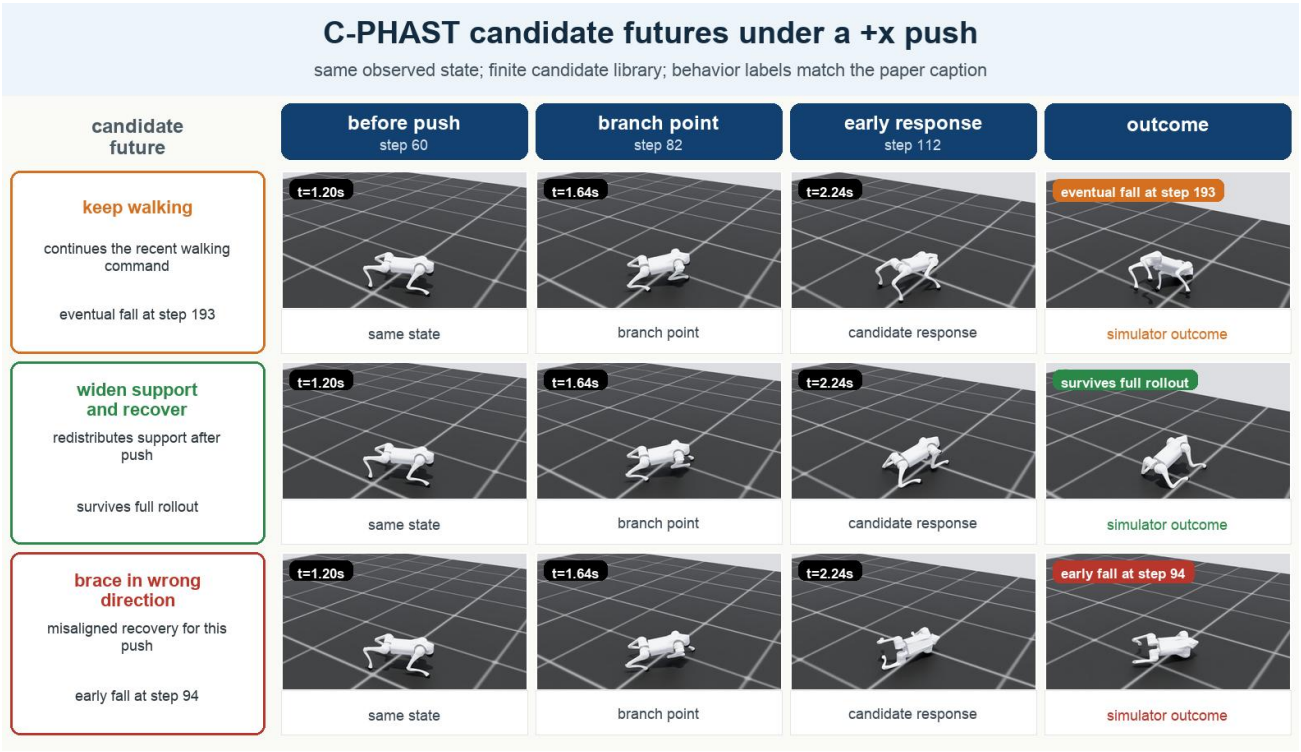

Figure 10. **Candidate futures from the same observed Go2 state.** After a $+x$ push on low-friction ground, we branch from the same observed state and execute three externally supplied candidate futures in Isaac Lab. Smooth continuation (*keep walking*) eventually falls, a support-redistributing future (*widen support and recover*) survives the full rollout, and a physically misaligned brace fails early. This is the Isaac Lab realization of the candidate-future interface introduced in Figure 1 and formalized in Figure 4: C-PHAST rolls out and scores candidate physical futures through typed pH readouts — each candidate a controlled flow ranked among the feasible ones by the decision objective of Eq. (1) — while the decision layer supplies the candidate recovery motions.

Together, the warning sweep and same-state branch show that the rollout factors carry enough physical signal to warn before simulator-level failure and sometimes change the selected action. The external candidate generator, envelope thresholds, and controller layer determine how that warning becomes a recovery maneuver.

4.3. Real Spot-with-Arm Physical-Consequence Forecasting

Deployment stress (Fig. 3): Q-ONLY SENSING on real hardware, plus the off-axis DECISION-FACING SCORING interface. Spot is the hardware-data check for the same observables-to-consequences-to-cards interface. At the hierarchy level of Table 6, this result sits at the interface-block level: base motion, leg support/contact, arm motion/contact, camera/depth sensing, and battery/power accounting own the typed readouts and planner factors. The learned query maps recent telemetry and candidate behavior descriptors into those named consequences; promoting the same hardware hierarchy to closed-loop PHASTCore rollout is the next hardware step. Each row is a real logged query: a two-second observed history, a proposed five-second behavior window, and the future physical consequences realized after that behavior in the teleoperated run. We cut six usable Spot-with-arm ROS-bag episodes into 1,070 overlapping windows, with 897 for training and 173 held out.

Figure 11 is the anchor for the subsection. It shows what the model sees, what future is being queried, and how the resulting consequences become action cards.

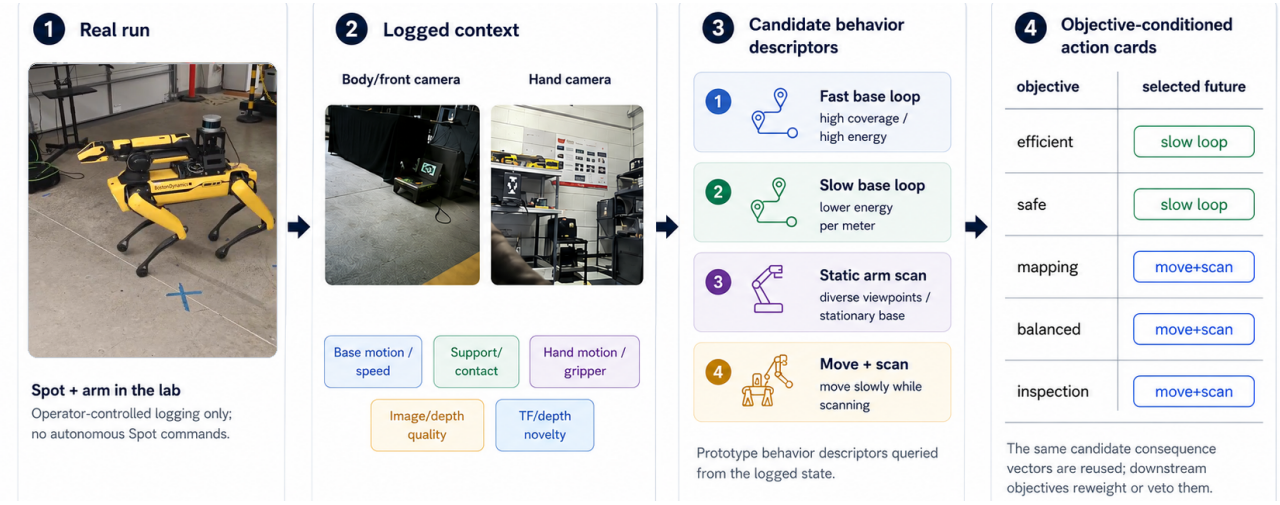

Figure 11. **Real Spot-with-arm shadow-mode query.** The hardware-data result uses logged Spot-with-arm runs: recent telemetry and synchronized camera summaries form the observed history, while the candidate side is a short behavior window such as a fast base loop, a slow base loop, a static arm scan, or a move-and-scan behavior. C-PHAST predicts named physical consequences from this query, and the decision objective of Eq. (1) turns those consequences into action cards through its factor weights. The experiment is offline/shadow-mode; C-PHAST does not command Spot in these runs.

Read the figure from left to right. Panels 1–2 are the observed history: recent base motion, contact/support, hand and gripper state, camera/depth quality, and TF/depth novelty summaries. Panel 3 is the candidate future being evaluated. In logged replay, some behavior-descriptor columns summarize what the robot actually executed; in deployment, the same fields would be supplied by a primitive library, planner, or forecast before execution. For each candidate, C-PHAST predicts named consequences such as motion, joint work, battery/current draw, support/contact, arm/hand motion, image quality, collision/contact proxies, and map-novelty proxies. Appendix J.1 gives the full dataset table and input-contract audit.

The ranking calculation is deliberately simple. Predicted consequences are first converted into planner-facing factors such as progress, work, battery draw, support loss, sensing quality, collision/contact residual, and depth/TF novelty. Each factor is scaled by its training-split 10–90 percentile range, and a fixed objective vector then assigns positive weights to desired consequences and negative weights to costs or risks. This produces one higher-is-better card score for each objective. Pearson correlation measures agreement between predicted and realized card scores; Spearman correlation measures whether the predicted cards preserve the high-to-low ordering of held-out query windows. For the representative same-history query in the figure, the selected future is the highest-scoring candidate under that objective, so efficient and safe cards favor a slow loop, while mapping, balanced, and inspection cards favor move-and-scan behavior.

With that interpretation, the held-out numbers are the hardware evidence for the interface. Typed consequences followed by objective weights reach mean Pearson/Spearman $0.880/0.860$ across efficiency, safety, mapping, balance, and inspection objectives, with per-objective correlations ranging from 0.744–0.948 and 0.753–0.939. The ablations explain why both sides of the query matter: history-only gives $0.767/0.737$, behavior-descriptors-only gives $0.780/0.778$, and history plus descriptors gives $0.880/0.860$. A direct scalar-objective head is rank-competitive ($0.871/0.875$), but it returns only a score. The typed interface preserves scalar ranking quality while exposing work, support/contact, sensing, collision, and residual channels for explanations, hard envelopes, and operator-facing action cards. Table 10 summarizes the main real-world checks.

Table 10. **Real-world Spot physical-consequence forecasting.** Spot supplies the hardware-data result for the typed consequence and action-card interface, using logged robot telemetry and behavior descriptors collected under human teleoperation. Pearson measures card-score agreement; Spearman measures ranking agreement.

Check	Result	Meaning
Consequence-to-card scoring	Mean Pearson/Spearman $0.880/0.860$ on held-out windows; per-objective Pearson 0.744–0.948	Real telemetry supports physically named future-consequence forecasting that remains useful for card ranking.
Input contract	History-only $0.767/0.737$, descriptors-only $0.780/0.778$, history plus descriptors $0.880/0.860$	Both the observed robot state and the proposed future matter.
Typed versus scalar score	Direct scalar head $0.871/0.875$ mean Pearson/Spearman	Typed readouts preserve ranking quality while exposing physical reasons.
Objective switch	Same observed history: efficient/safe favor slow-loop; mapping/balanced/inspection favor move-and-scan	The highest-scoring candidate changes when the same consequences are reweighted.
Episode holdout	Leave-one-episode-out mean Pearson/Spearman $0.652/0.545$	Signal remains positive under the harder episode split; the window split is the main hardware-interface number.

Together, these checks make Spot the paper's real-world forecasting result: C-PHAST maps hardware telemetry and behavior descriptors to physically named future consequences and action cards. Spot verifies the observables-to-consequences-to-cards path on real robot telemetry; Isaac verifies same-state branch execution and action selection in simulation. Appendix J.1 states the Spot information contract, including which descriptors are decision-time quantities and which are logged replay proxies, and Appendix Tables 39–44 give the factor construction, objective weights, per-objective results, ablations, and scalar-head comparison.

4.4. Few-Shot One-Step Adaptation

Deployment stress (Fig. 3): target-data efficiency, an axis orthogonal to the specification-withholding columns. The cross-robot table leaves a practical question open. If target Go2 data become available, can an unconstrained recurrent model quickly recover the missing transfer structure, or does the pH bias still matter? We test this question in the easier one-step setting by fine-tuning pretrained models for 20 epochs (lr=10^{-4}) on $N_{\text{adapt}} \in \{5, 10, 25, 50\}$ Go2 trajectories, then testing on held-out trajectories from *both* robots.[2] We keep this test one-step because the point is target-data efficiency, not another rollout table. It isolates whether a small amount of Go2 supervision can overwrite source-specific recurrent dynamics.

Table 11. Few-shot cross-robot *one-step* adaptation. For one-step prediction, zero-shot C-PHAST remains better than Shared-GRU adapted with 50 target trajectories ($\mathbf{1.7}\times$). One-step MSE ($\times 10^{-4}$); no rollout metric is reported here. Models pretrained on ANYmal-D, fine-tuned on N_{adapt} Go2 trajectories. Mean $\pm$ std over 5 seeds. Best per row in **bold**.

	C-PHAST (22,862)		**Shared-GRU** (60,804)	
N_{adapt}	Go2 ↓	AD ↓	Go2 ↓	AD ↓
0 (zero-shot)	$\mathbf{9.8 \pm 0.1}$	4.8 ± 0.0	25.5 ± 1.0	$\mathbf{4.4 \pm 0.4}$
5	$\mathbf{9.6 \pm 0.1}$	$\mathbf{4.8 \pm 0.0}$	20.0 ± 1.2	7.8 ± 1.2
10	$\mathbf{9.6 \pm 0.1}$	$\mathbf{4.8 \pm 0.0}$	19.1 ± 1.2	7.9 ± 1.2
25	$\mathbf{9.6 \pm 0.1}$	$\mathbf{4.8 \pm 0.0}$	19.1 ± 1.3	8.1 ± 1.2
50	$\mathbf{9.5 \pm 0.1}$	$\mathbf{4.8 \pm 0.0}$	17.0 ± 1.3	8.2 ± 1.3

Table 11 shows an informative asymmetry. Shared-GRU does use target data: its Go2 one-step error improves by 33% at N=50, demonstrating the capacity of its unconstrained hidden state. But adaptation does not erase the transfer gap. Even after 50 Go2 trajectories, Shared-GRU's target error (17.0×10^{-4}) remains $1.7\times$ worse than C-PHAST's zero-shot error (9.8×10^{-4}). For this one-step metric, the port-Hamiltonian inductive bias encodes shared physical structure that limited target-data gradient descent does not recover.

[2] We reuse the same matched-protocol source checkpoints as the main Isaac transfer experiment: both models are pretrained on 350 ANYmal-D trajectories for 50 epochs with one-step MSE only, then adapted on Go2. Results are reported over 5 seeds (42–46). Full details in Appendix I.3.1.

C-PHAST improves only marginally with target data (-3% at $N{=}50$). That is the expected pattern if most transferable structure has already been captured and the remaining gap is dominated by robot-specific contact and actuator effects outside the q-only model class.

Catastrophic forgetting. The source-robot side of the table gives the second payoff. C-PHAST exhibits essentially no source degradation: ANYmal-D error remains at 4.8×10^{-4} across all N_{adapt}. Shared-GRU shows clear catastrophic forgetting: source error rises from 4.4 to 8.2 at $N{=}50$ ($+86\%$). The pH constraints act as an implicit regularizer by keeping adaptation tied to the learned energy, dissipation, and port structure.

4.5. Variable-N Generalization

Deployment stress (Fig. 3): LAYOUT/COUNT—the densest, contested demand column. After fixed-layout cross-robot transfer, the next stress changes how many repeated parts the deployed system contains. We ask whether a model trained at one subsystem count N can deploy at others without retraining.

Robot arm. On the robot-arm benchmark (Appendix G), models are trained on $N{=}4$ and evaluated without retraining on $N \in \{2, 3, 4, 5, 6, 8\}$. Fixed-dimension baselines (S5, LRU, LinOSS, GRU) are structurally incompatible with $N \neq 4$ due to hardcoded input projections (Table 24). Table 12 shows takeover rollout ($H{=}50$): C-PHAST remains stable across all N, while MPNN diverges catastrophically at $N{=}8$. The companion one-step table is in Appendix G.2.

Table 12. Open-loop stability on robot-arm dynamics (q-only). C-PHAST remains stable across all tested N, while MPNN fails at the largest extrapolation. Takeover rollout error ($\text{MSE}^{\text{take}}(H{=}50)$ with history length $L_h{=}5$). Train on $N{=}4$, test on all N. Mean $\pm$ std over seeds. Best per column in **bold**.

Model	Params	N=2	N=3	N=4	N=5	N=6	N=8
C-PHAST (known coupling)	13,015	**0.0029 ± 0.0001**	**0.0021 ± 0.0000**	**0.0017 ± 0.0000**	**0.0016 ± 0.0000**	**0.0018 ± 0.0000**	**0.0014 ± 0.0000**
C-PHAST (partial coupling)	21,977	0.0031 ± 0.0001	0.0022 ± 0.0001	0.0018 ± 0.0001	0.0018 ± 0.0000	0.0019 ± 0.0000	**0.0014 ± 0.0000**
C-PHAST (learned skew, one-step)	23,386	0.3208 ± 0.0008	0.3191 ± 0.0012	0.2920 ± 0.0012	0.2474 ± 0.0015	0.2961 ± 0.0012	0.2943 ± 0.0003
C-PHAST (learned skew, rollout-aligned)	23,386	0.0684 ± 0.0038	0.0624 ± 0.0075	0.0645 ± 0.0053	0.0528 ± 0.0071	0.0663 ± 0.0046	0.0584 ± 0.0044

MuJoCo legged robots. C-PHAST is not being asked here to transfer between arbitrary robot bodies; it is being re-instantiated across a homologous MuJoCo family with the same 2-DOF leg template and varying leg count. Table 13 reports takeover rollout ($H{=}50$), which is the relevant stress test for this homologous leg-count transfer. C-PHAST achieves 3–7$\times$ lower rollout error than the nearest stable baseline (Shared-GRU: 0.013–0.020 vs. C-PHAST: 0.002–0.005). Figure 17 (Appendix) visualizes the corresponding MuJoCo leg-count transfer qualitatively. MPNN is rollout-unstable (1/3 seeds diverges per morphology). The coupling ablations have nearly identical error, confirming that the primary driver of morphology transfer is the shared per-leg dynamics model rather than the coupling parameterization. On this benchmark, coupling structure functions mainly as a correctness constraint; the robot-arm benchmark is the setting where coupling parameterization materially affects accuracy (Appendix G.3).

Table 13. MuJoCo rollout at $H{=}50$ with history length $L_h{=}5$. The main signal is rollout robustness rather than one-step margin: C-PHAST achieves up to 7$\times$ lower error than the nearest stable baseline. Mean $\pm$ std over 3 seeds.

Model	Params	Test (N=2)	Test (N=4)	Test (N=6)
C-PHAST	22,860	$\mathbf{0.002 \pm 0.000}$	$\mathbf{0.005 \pm 0.000}$	$\mathbf{0.003 \pm 0.000}$
Shared-GRU	60,163	0.013 ± 0.002	0.014 ± 0.002	0.020 ± 0.009
DeepSets	25,859	*unstable*†	0.14 ± 0.07	0.26 ± 0.27
MPNN	42,178	*unstable*†		

†1 of 3 seeds diverges ($> 10^3$); remaining seeds bounded.

4.6. Coupled FitzHugh–Nagumo Excitable Circuits

Deployment stress (Fig. 3): LAYOUT/COUNT, outside robotics. The robotics count-transfer benchmarks test repeated mechanical blocks. To check that the same mechanism is not robot-specific, we also evaluate C-PHAST on diffusively coupled FitzHugh–Nagumo (FHN) units, a canonical excitable electrical-circuit model (FitzHugh, 1961) with broader use in

excitable systems, including cardiac and electronic pattern-forming models (Cebrián-Lacasa et al., 2024). In plain terms, each FHN unit is a small two-state excitable element: a voltage-like state rises, a recovery state brings it back down, and neighboring cells influence each other through diffusive voltage coupling. This benchmark is deliberately narrower than the cardiac-field study discussed in the project notes: here the paper claim is only repeated-block recomposition for coupled excitable circuits, not real cardiac data or bidomain readout modeling.

Setup. Each FHN unit is represented as a 1-DOF pH circuit block with voltage/recovery state and a cubic tunnel-diode nonlinearity. This block is *active*—the tunnel-diode term injects energy—and the diffusive inter-cell coupling is dissipative rather than skew, so the benchmark tests repeated typed-block recomposition and is *not* an instance of the skew, power-preserving guarantee of Theorem 3.1 (the energy ledger is derived in Appendix H.2). We couple N identical units in a chain with diffusive voltage coupling strength $g{=}0.1$. C-PHAST reuses one shared excitable-cell block and rebuilds the graph Laplacian at deployment. We train on $N{=}3$ coupled neurons (200 trajectories, 200 steps, $\Delta t{=}0.1$) and evaluate without retraining on $N \in \{2, 3, 4, 5, 6, 8\}$. The N-PHDAE baseline follows the same compose-and-predict protocol: train a single-neuron model and compose copies with the known graph at test time (Neary et al., 2024). Appendix H.2 records the protocol and interpretation boundary.

Results. Table 14 reports the benchmark in two views. The top block checks fixed-$N{=}3$ accuracy across knowledge regimes before any count transfer is attempted. The exact KNOWN rows are best, as expected, because the local FHN physics and coupling are specified analytically; increasing the integration substeps from $L{=}1$ to $L{=}4$ reduces one-step MSE from $4.0{\times}10^{-8}$ to $5.5{\times}10^{-10}$. Among learned rows, N-PHDAE is the strongest raw forecaster at fixed $N{=}3$ ($8.0{\times}10^{-7}$ one-step MSE), while C-PHAST PARTIAL is an order of magnitude higher ($8.2{\times}10^{-6}$) because it learns the constitutive nonlinearity through the pH block pipeline. The fully learned C-PHAST LAW-UNKNOWN row is much weaker in absolute rollout error, so we use it as a diagnostic row rather than as the main FHN claim.

The bottom block is the actual recomposition test. After training at $N{=}3$, the learned cell block is copied and the diffusive graph is rebuilt for new cell counts. C-PHAST PARTIAL transfers from $N{=}3$ to unseen $N \in \{2, 4, 5, 6, 8\}$ with essentially flat one-step error (6.6–$8.2{\times}10^{-6}$), and the LAW-UNKNOWN row is likewise flat around $2{\times}10^{-3}$. Thus FHN plays a specific role in the evidence graph: it confirms that repeated typed-block transfer is not restricted to robot joints or legs, while the DC microgrid benchmark below tests the harder heterogeneous node-edge case.

Table 14. **Coupled FHN: fixed-N hierarchy and repeated-block transfer.** *Top*: fixed $N{=}3$ accuracy across C-PHAST knowledge regimes, with N-PHDAE compose-and-predict as the learned electrical baseline. *Bottom*: C-PHAST transfer from $N{=}3$ to unseen neuron counts by copying the shared excitable-cell block and rebuilding the chain coupling. Mean over 3 seeds, 100 epochs; stds are small for the table-scale means and are reported in the artifact. Bold entries mark the best value within each table block/metric.

Model ($N{=}3$)	**1-step MSE** ↓	$H{=}10$ ↓	$H{=}50$ ↓
C-PHAST KNOWN ($L{=}4$)	$\mathbf{5.5{\times}10^{-10}}$	$\mathbf{1.9{\times}10^{-8}}$	$\mathbf{3.1{\times}10^{-7}}$
C-PHAST KNOWN ($L{=}1$)	$4.0{\times}10^{-8}$	$9.5{\times}10^{-7}$	$5.8{\times}10^{-6}$
N-PHDAE (composed)	$8.0{\times}10^{-7}$	$1.4{\times}10^{-5}$	$5.8{\times}10^{-5}$
C-PHAST PARTIAL	$8.2{\times}10^{-6}$	$1.0{\times}10^{-3}$	$1.3{\times}10^{-3}$
C-PHAST LAW-UNKNOWN	$2.0{\times}10^{-3}$	$7.9{\times}10^{-1}$	$1.2{\times}10^{1}$

	Compositional transfer: 1-step MSE ↓					
Model	$N{=}2$	$N{=}3$ (train)	$N{=}4$	$N{=}5$	$N{=}6$	$N{=}8$
C-PHAST PARTIAL	$\mathbf{6.6{\times}10^{-6}}$	$\mathbf{8.2{\times}10^{-6}}$	$\mathbf{7.5{\times}10^{-6}}$	$\mathbf{6.7{\times}10^{-6}}$	$\mathbf{6.7{\times}10^{-6}}$	$\mathbf{6.6{\times}10^{-6}}$
C-PHAST LAW-UNKNOWN	$2.0{\times}10^{-3}$	$2.0{\times}10^{-3}$	$2.1{\times}10^{-3}$	$1.9{\times}10^{-3}$	$2.1{\times}10^{-3}$	$2.0{\times}10^{-3}$

4.7. Mechanism Checks for Compositional Transfer

The repeated-block results leave two alternative explanations. One is that local weight sharing alone is enough: any model copied across parts might transfer. Another is that the result depends only on the local pH block, not on the deployed composition rule. This subsection is therefore a mechanism audit rather than a new benchmark.

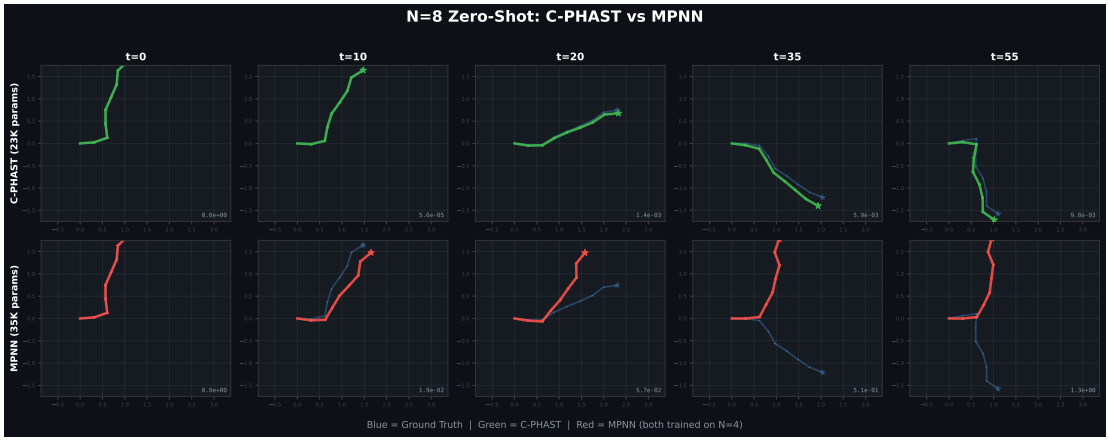

Figure 12. **Zero-shot transfer at $N{=}8$: C-PHAST vs MPNN** (both trained on $N{=}4$). The hard extrapolation case makes the local-sharing failure mode visible: C-PHAST (green) stays close to ground truth with MSE $9.8{\times}10^{-3}$ at $t{=}55$, while the locally shared MPNN baseline (red) diverges to MSE 1.3 (130× gap).

Figure 12 gives the visible answer to the first alternative: local sharing and graph message passing do not by themselves prevent hard-extrapolation failure. The remaining checks are narrower diagnostics. Appendix G.3 varies the unknown coupling parameterization on robot arms; those rows show that coupling choices affect the bounded-error regime, but they are not the main explanation for every benchmark. Appendix G.6 compares fixed-dimension SSM baselines at $N{=}4$, Appendix G.7 verifies near-zero skew-coupling power residuals, Appendix G.8 tests coupling normalization, Appendix G.9 compares configuration-dependent $\hat{\mathbf{C}}(q)$ with constant $\hat{\mathbf{C}}$, and Appendix G.10 perturbs the robot-arm mass prior. Together these checks support the transfer interpretation used above: the reusable local pH block and the deployed composition rule are doing work, while the exact strength of each mechanism depends on the domain and protocol.

4.8. DC Microgrid: Mixed Node-Edge Composition

Deployment stress (Fig. 3): TOPOLOGY CHANGES and Q-ONLY SENSING—the columns C-PHAST is designed to address. A key question beyond the repeated-subsystem robot and FHN benchmarks is whether C-PHAST can compose *heterogeneous* block types. A DC microgrid is a direct-current power network in which converter-controlled sources or storage units feed local loads through electrical lines. A DC microgrid gives that test in domain language: converter nodes and transmission-line edges remain the physical assets, but deployment can change when DGUs are added or removed, active feeders are rewired, or the protection layer observes only bus voltages. The fixed object is the DGU/line library; the redeployed object is the graph and observation interface.

We test this setting on a power-electronics benchmark derived from Cucuzzella-style DC microgrid regulation (Cucuzzella et al., 2019). Operationally, the reconfiguration tier is a stylized feeder-reconfiguration problem: after a protection or overload event, the operator changes the active tie-switch pattern while keeping the same DGU and line assets in service (Tucci et al., 2018; Sadabadi et al., 2018). The voltage-only tier mirrors protection-style observations, where bus voltages remain available but line currents and converter states may not be exposed to the deployed model (Liu et al., 2018). Figure 2, introduced in the opening motivation, gives this digital-twin redeployment picture visually; the paragraphs below turn it into concrete protocols.

Setup. The microgrid consists of N Distributed Generation Units (DGUs) connected by M transmission lines. DGUs are node blocks and lines are edge blocks. Each DGU is a 1-DOF pH subsystem with charge $q{=}CV$ and flux $p{=}LI_t$ driven by a Buck converter voltage u_i and load current I_{Li}. Each transmission line is a separate 1-DOF pH subsystem with inductive storage and series line resistance (an RL line). The two block types are coupled via an oriented incidence matrix B_{inc}: DGU q-equations receive line currents ($B_{\text{inc}} \cdot I_{\text{line}}$) and line p-equations receive voltage differences ($-B_{\text{inc}}^{\top} V$). This is a node-edge coupling rather than the chain-style repeated-block coupling used in the robotics benchmarks. C-PHAST treats DGUs and lines as two distinct typed blocks, each with shared weights and known graph interconnection. We use the Neary benchmark parameters (Neary et al., 2024): $R_t{=}1.2\,\Omega$, $L_t{=}1.8\,\text{H}$, $C_t{=}2.2\,\text{F}$, with heterogeneous line parameters sampled from $\mathcal{U}(0.1, 2.0)$. Time-varying control inputs $u_i(t)$ and load currents $I_{Li}(t)$ are passed through the compositional action pathway (half-kick splitting).

We train on $N{=}3$ (100 trajectories, 100 steps, $\Delta t{=}0.01$) and transfer to $N \in \{2, 3, 4, 5, 8\}$ by copying shared DGU and line core weights, rebuilding the incidence matrix for each test graph. N-PHDAE uses the same compose-and-predict protocol: train a single-DGU model, compose N copies with known line dynamics.

The deployment stresses are then simple changes to this setup. In the full-state known-graph protocol, the model sees the complete state and changes only the number of DGU/line blocks. In the topology-reconfiguration protocol, $B_{\text{inc}} \to B'_{\text{inc}}$ rewires the node-edge coupling terms while the learned DGU and line cores remain fixed. In the voltage-only protocol, topology is still declared but bus voltages are the only online measurements; line currents and converter states become hidden deployment variables.

In addition to rollout MSE, the microgrid tables report design-facing diagnostic errors: Vbal err is the RMS error in the Cucuzzella-style voltage-balancing residual, i.e., the weighted-average bus-voltage residual relative to the weighted-average voltage references (Cucuzzella et al., 2019). Because the maintained benchmark does not include per-DGU capacity ratings, we use equal weights, so this reduces to arithmetic-average voltage balancing. Share err is the RMS error in the proportional-current-sharing residual $WI_t - \overline{WI_t}\mathbf{1}$, again with equal weights in the reported benchmark. $|\Delta H|$ err is absolute error in Hamiltonian energy change over the prediction step/window. The API still records energy-throughput efficiency, voltage-deviation, and coefficient-of-variation diagnostics, but these are auxiliary diagnostics rather than the regulation/current-sharing metrics foregrounded in the paper. All reported diagnostic errors are lower-is-better.

Table 15 is the regime map for the results. It should be read as a boundary map, not as a claim that C-PHAST uniformly dominates the circuit-specialized baseline.

Table 15. **Microgrid regime summary.** The microgrid evidence is regime-dependent. Full numeric tables are in Appendix H.4: Table 28 for full-state typed-block composition, Table 30 for topology reconfiguration, and Table 31 for voltage-only sensing.

Deployment regime	What changes	Main reading	Where to inspect
Full-state known graph	DGU/line count changes; all state channels and topology are provided.	DGU/line blocks compose across graph sizes, but N-PHDAE is stronger on rollout metrics.	Table 28
Topology reconfiguration	The feeder graph is rewired at test time, and the new B'_{inc} is provided.	C-PHAST PARTIAL gives tighter design diagnostics, even when N-PHDAE is stronger on raw rollout.	Table 30
Voltage-only sensing	Topology is provided, but line currents and converter states are hidden.	No decisive single-model winner: the stable models remain close, with different winners by metric.	Table 31
Sensing ladder	Current channels are removed progressively.	Degradation follows the sensing tiers; the C-PHAST LAW-UNKNOWN variant breaks earlier.	Table 32, Figure 18

Results. The microgrid evidence should be read as four linked points rather than as a uniform numerical win.

Heterogeneous typed-block composition. This benchmark shows that C-PHAST composes two different subsystem types, DGUs and transmission lines, under one typed-block port-Hamiltonian template. The same learned per-type cores are reused across graph sizes, while only the incidence coupling is rebuilt. Within the C-PHAST rows, transfer to new N remains stable under this typed-block recomposition.

Full-state known-graph prediction remains stronger for N-PHDAE. The updated maintained artifact no longer supports using this benchmark as a blanket numerical win over N-PHDAE. At the training size $N{=}3$, C-PHAST PARTIAL reaches the lowest *one-step* MSE in Appendix Table 28 ($2.4{\times}10^{-7}$), but N-PHDAE remains numerically stronger across graph sizes on the more demanding rollout metrics. We therefore treat the full-state known-graph protocol primarily as a demonstration of heterogeneous typed-block composition under a unified pH template, not as the microgrid setting in which C-PHAST is most competitive.

Deployment-stressed settings are more informative for this domain. The intended microgrid use case is not only to fit a known circuit, but to keep a structured surrogate useful when the deployed feeder or sensing interface changes. We therefore focus the main text on the two follow-up protocols that best match that use case.

Topology reconfiguration. This protocol asks whether explicit wiring helps when the feeder graph is rewired at test time. Here C-PHAST PARTIAL remains stable and is a tighter design surrogate than N-PHDAE on voltage balancing, ΔH, and

current-sharing residual, even though N-PHDAE remains stronger on raw rollout MSE; Appendix Table 30 reports the full setting. The maintained rerun leaves this ranking unchanged.

Voltage-only partial sensing. This protocol asks whether regulation diagnostics survive when currents and converter states are hidden. In this hardest sparse-sensing tier, the finished 5-seed ladder no longer supports a decisive single-model ranking between C-PHAST PARTIAL and N-PHDAE on raw rollout. C-PHAST PARTIAL attains the best one-step and hidden-state errors, C-PHAST KNOWN attains the lowest $H{=}50$ observed-rollout error, and N-PHDAE is narrowly best on Vbal and $|\Delta H|$. The useful scientific point is therefore not a blanket numerical win, but that the three stable models remain close under sparse sensing while the C-PHAST LAW-UNKNOWN variant remains substantially worse on rollout or voltage balancing. Appendix Table 31 reports this setting. Appendix Table 32 reports the full sensing ladder from full state to bus voltages only: C-PHAST KNOWN, C-PHAST PARTIAL, and N-PHDAE stay close across tiers, while the C-PHAST LAW-UNKNOWN variant is already poor on design metrics and deteriorates on sparse-rollout errors.

Typed residuals identify the channel of deployment mismatch. The forecast table alone hides a capability that matters for state-estimation and protection-style monitoring. N-PHDAE remains the strongest raw forecaster in several microgrid rows, but C-PHAST exposes the physical channel through which the composed surrogate stops matching the deployment. For a residual channel c, define the reference-normalized amplification $A_c(s) = r_c(s)/(r_c(\mathrm{ref}) + \epsilon)$ using the C-PHAST PARTIAL reference chain as the nominal case. Table 16 reports three channels that are most diagnostic under topology and DGU-count changes. The removed-DGU case raises energy-balance, line-current, and interconnect-power residuals by $183\times$, $42\times$, and $43\times$ respectively, whereas the star rewire remains near reference on all three. Thus the contribution is not that C-PHAST replaces N-PHDAE as the circuit-specialized predictor; it is that the same typed rollout object provides a compositional consistency monitor.

Table 16. **Typed residual amplification under microgrid reconfiguration.** Values are C-PHAST PARTIAL residuals normalized by the same channel on the reference chain, $A_c(s) = r_c(s)/(r_c(\mathrm{ref}) + \epsilon)$. N-PHDAE is included only to show the raw-forecasting boundary: it remains the strongest circuit-specialized forecaster in these rows, while C-PHAST exposes which physical channel changed.

Scenario	C-PHAST $H{=}50$	N-PHDAE $H{=}50$	Energy excess	Line current	Interconnect power / diagnosis
Reference chain	$\mathbf{3.71\times10^{-5}}$	6.49×10^{-5}	$1.0\times$	$1.0\times$	$1.0\times$; nominal typed residual scale
Removed DGU	3.00×10^{-2}	$\mathbf{7.64\times10^{-5}}$	$183\times$	$42\times$	$43\times$; missing object / coupling mismatch
Added DGU	2.53×10^{-3}	$\mathbf{7.69\times10^{-5}}$	$26\times$	$5.4\times$	$4.0\times$; new object / coupling load
Ring rewire	3.79×10^{-3}	$\mathbf{5.82\times10^{-5}}$	$37\times$	$7.0\times$	$5.9\times$; topology/interconnect mismatch
Star rewire	$\mathbf{4.00\times10^{-5}}$	6.48×10^{-5}	$0.94\times$	$0.94\times$	$0.86\times$; near-reference response

Across all benchmarks, the same typed-block pH template supports transfer, although the strength of the numerical gains is domain-dependent. Section 5 discusses the PHAST-style control template that remains compatible with this compositional setting.

5. Control Compatibility via Port-Hamiltonian Controller Blocks

The experiments above treat C-PHAST as a rollout model whose typed consequences are consumed by planners, critics, and protection logic. A second downstream use is more structural: because C-PHAST keeps the learned plant in port-Hamiltonian form, it also preserves the power variables used by passivity-based controllers. In that view a controller is not an afterthought attached to a black-box predictor; it can be another pH block whose objective is encoded in its own storage function and whose ports interconnect with the learned plant through power-conjugate variables. Classical IDA–PBC and Energy–Casimir control exploit this viewpoint by choosing a desired storage/interconnection/dissipation structure so the shaped storage becomes a Lyapunov function (Ortega et al., 2002; 2008). This is an instance of *control by interconnection*, whose achievable closed-loop interconnection structures and Casimir functions are characterized classically by Cervera et al. (2007); recent work carries it to optimization- and MPC-based synthesis for nonlinear pH plants (Gernandt et al., 2026), and to passivity-based stabilization by dissipative feedback in the infinite-dimensional boundary-control setting (Le Gorrec et al., 2005; Augner, 2019), including networks of interconnected pH subsystems (Augner, 2020). We do not develop controller synthesis against that literature. The scope of this section is interface compatibility: C-PHAST preserves the pH ports needed by such controller blocks, and the planar-arm Energy–Casimir experiment inherited from PHAST (Bhardwaj & Bajaj, 2026) validates one concrete attachment. This is complementary to the model-predictive critic interface in Section 3.6: the critic uses C-PHAST rollouts to rank candidate futures, whereas the controller block uses the same pH port variables to synthesize a storage-shaping feedback law.

Generic pH controller interface. At the level needed here, the learned plant must expose a power port: an input through which the controller can do work and an output that reports the conjugate flow. Write that plant block as

$$\dot{x}_p = (J_p - R_p)\nabla H_p(x_p) + \mathbf{B}_p u_p, \qquad y_p = \mathbf{B}_p^\top \nabla H_p(x_p), \tag{38}$$

where u_p and y_p are power-conjugate plant input/output variables, so $y_p^\top u_p$ is the power supplied through the port. A pH controller block must expose the same kind of pair. With internal state ξ and reference or objective parameter r, write it as

$$\dot{\xi} = (J_c - R_c)\nabla_\xi H_c(\xi; r) + \mathbf{B}_c u_c, \qquad y_c = \mathbf{B}_c^\top \nabla_\xi H_c(\xi; r). \tag{39}$$

The objective is carried by H_c (or, in IDA–PBC language, by the desired closed-loop Hamiltonian H_d), while damping and interconnection choices determine how energy is routed or removed. For the collocated case used below, connect controller output to plant input and plant output to controller input, with a separate damping injection v:

$$u_p = -y_c + v, \qquad u_c = y_p, \qquad v = -K_{\text{inj}} y_p, \quad K_{\text{inj}} \succeq 0 \tag{40}$$

The port power of the connected pair then satisfies the balance

$$y_p^\top u_p + y_c^\top u_c = y_p^\top v = -y_p^\top K_{\text{inj}} y_p \leq 0. \tag{41}$$

Thus the plant-controller connection is power-preserving except for the explicitly injected damping. This is the control-side payoff of preserving pH ports: the same variables that make C-PHAST compositional also provide attachment points for pH controller blocks. Figure 13 depicts the Energy–Casimir instance used in our experiments.

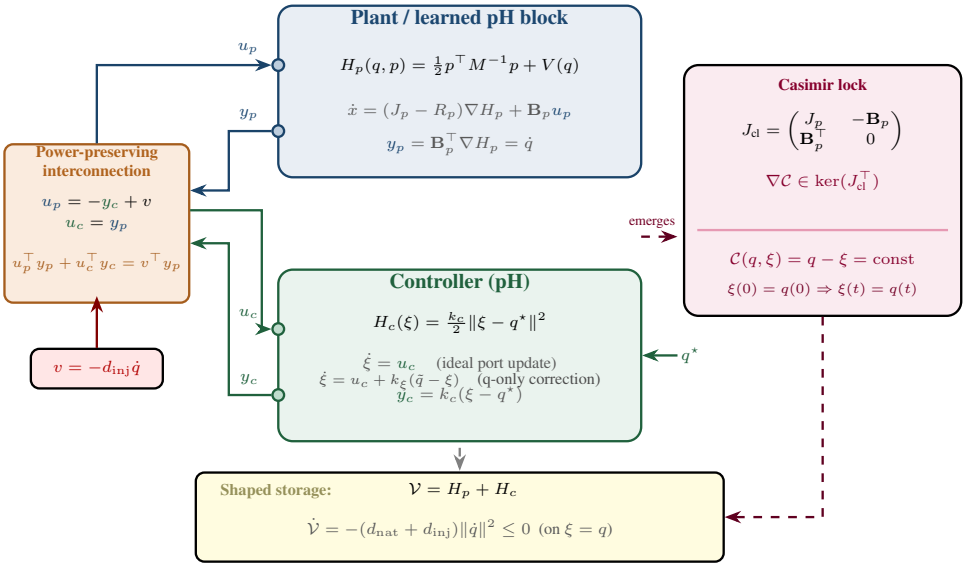

Figure 13. **A pH controller-block attachment: Energy–Casimir as the demonstrated instance.** The same pH ports used by a C-PHAST block can attach a pH controller block through a power-preserving interconnection. The Casimir relation $\mathcal{C}(q,\xi) = q - \xi$ locks the controller state to the plant configuration in the ideal exact-port setting, yielding the shaped storage inequality used in Section 5. This is a control-compatibility template, not the main learned-dynamics algorithm; other pH/IDA–PBC-style controller blocks can use the same interface when their matching conditions are satisfied.

Planar-arm Energy–Casimir instance. For the planar-arm benchmark, the plant is mechanical with $x_p = (q,p)$, Hamiltonian

$$H_p(q,p) = \frac{1}{2}p^\top M^{-1}p + V(q), \qquad \mathbf{B}_p = \begin{bmatrix} 0 \\ I \end{bmatrix}, \qquad y_p = \mathbf{B}_p^\top \nabla H_p = \dot{q}.$$

This is the scalar, isotropic specialization of the generic damping-injection matrix above: in the arm experiment we set $K_{\text{inj}} = d_{\text{inj}} I$, so $v = -K_{\text{inj}} y_p$ becomes $v = -d_{\text{inj}}\dot{q}$. The controller state $\xi \in \mathbb{R}^N$ stores a shaped potential around the target configuration $q^\star$:

$$H_c(\xi) = \frac{k_c}{2}\|\xi - q^\star\|^2, \qquad y_c = \nabla_\xi H_c = k_c(\xi - q^\star), \tag{42}$$

with controller dynamics $\dot{\xi} = u_c$. The output y_c is the generalized force that pulls the controller state toward $q^\star$; once the interconnection locks ξ to the plant configuration, this term reshapes the plant potential around the target. Choosing the interconnection

$$u_p = -y_c + v, \qquad u_c = y_p, \qquad v = -d_{\text{inj}}\dot{q} \tag{43}$$

makes $\dot{\xi} = \dot{q}$ in the ideal exact-port setting and therefore preserves the Casimir relation $q - \xi$. With initialization $\xi(0) = q(0)$, the Casimir constraint $\xi(t) = q(t)$ is preserved, and the **shaped Lyapunov function**

$$\mathcal{V}(q, p, \xi) = H_p(q, p) + H_c(\xi) = \frac{1}{2} p^\top M^{-1} p + V(q) + \frac{k_c}{2}\|\xi - q^\star\|^2 \tag{44}$$

reduces on the Casimir manifold to

$$\mathcal{V}(q, p, q) = \frac{1}{2} p^\top M^{-1} p + V(q) + \frac{k_c}{2}\|q - q^\star\|^2.$$

Along this manifold the dissipation law becomes

$$\dot{\mathcal{V}} = -\dot{q}^\top (D_{\text{nat}} + d_{\text{inj}} I)\dot{q} \le 0, \tag{45}$$

where $D_{\text{nat}} \succeq 0$ is the plant's natural damping. Under the exact-observation and local-minimum assumptions stated in Theorem A.1, LaSalle's invariance principle yields local asymptotic stability of the equilibrium $(q^\star, 0, q^\star)$. On the Casimir manifold, the effective control law seen by the plant is

$$u_p = -k_c(q - q^\star) - d_{\text{inj}}\, \dot{q}, \tag{46}$$

where k_c shapes the potential and d_{inj} injects damping.

Compatibility across joint count (an interface check). The same exact-port mechanism that keeps C-PHAST compositional also keeps this controller attachment well defined as joints are added: for any deployed skew coupling $\hat{\mathbf{C}} = -\hat{\mathbf{C}}^\top$ the total coupling power vanishes ($\dot{q}^\top \hat{\mathbf{C}} \dot{q} = 0$)—an inherited port-Hamiltonian property, not a new control result—so the sum of the per-joint shaped storage functions obeys the same dissipation inequality. Appendix A, Theorem A.4, records this exact-port continuous-time statement, and the planar-arm experiments in Appendix E show that the same per-joint gains remain effective for $N \in \{2, 4, 8\}$ without retuning. We report this as a software/interface compatibility check—the exposed ports admit the classical Energy–Casimir block unchanged as the system is resized—not as a control-theoretic transfer contribution.

Q-only implementation. The robotics experiments mostly use the harder observation contract: positions are measured, while port velocities or momenta must be inferred from history. Under position-only sensing, the same controller must therefore be driven by a velocity estimate $\hat{\dot{q}}$ and an auxiliary controller state update rather than by exact port variables. The deployed update is therefore

$$\dot{\xi} = \hat{\dot{q}} + k_\xi(\tilde{q} - \xi), \qquad u_p = -k_c(\xi - q^\star) - d_{\text{inj}}\hat{\dot{q}}, \tag{47}$$

where $\tilde{q}$ is the measured position and $k_\xi > 0$ corrects drift of the approximate Casimir lock. This is the deployment caveat that connects the control section back to the observation contract: when $\hat{\dot{q}} \neq \dot{q}$, the damping term is no longer sign-definite and the exact continuous-time dissipation proof does not apply verbatim. We therefore use q-only control as an implementation-level compatibility test, while keeping the exact stability statement tied to exact port observations. Appendix E gives the online update details and the planar-arm evaluation. The main-text takeaway is that C-PHAST preserves the pH controller interface: the planar-arm experiment demonstrates one Energy–Casimir controller block, while IDA–PBC-style designs provide the broader class of storage-shaping controllers that can use the same ports when the relevant matching conditions are satisfied. The exact-port composition and controller compatibility used here are inherited port-Hamiltonian facts, not contributions of this paper; the distinct question of *robust* control on the *learned, redeployed* plant—where model, matching, observation, and discretization errors enter the closed-loop certificate—is a separate research direction, for which approximate/neural IDA–PBC matching already provides practical-stability guarantees on single learned plants (Sanchez-Escalonilla et al., 2024), that we leave to future work.

6. Conclusion

We introduced C-PHAST, a compositional port-Hamiltonian world model whose learned object is not a single fixed simulator, but a reusable physical contract. The contract keeps local block semantics fixed, rebuilds the declared composition for a target specification $\mathcal{S}_{\text{target}}$, and exposes rollouts through named ports, consequences, and planner-facing factors. Across the benchmarks studied here, the same idea supports subsystem-count transfer in arms and homologous legged robots, fixed typed-layout transfer from ANYmal-D to Go2, repeated excitable-cell recomposition in FHN circuits, and typed DGU/line recomposition in DC microgrids.

The main empirical lesson is consistent with port-Hamiltonian structure providing the principal transferable bias, while composition supplies the mechanism for re-instantiation across counts and layouts—not modularity by itself. In the cross-robot Isaac Lab comparison, port-Hamiltonian structure reduces the open-loop transfer gap from $21\times$ for a fixed-dimensional GRU to $3.7\times$ for monolithic PHAST, composition tightens it further to $1.6\times$, and composition without the pH constraint collapses outright (Table 8). The same table reports the refreshed rollout mode, and there the ordering sharpens rather than dissolves: re-anchored in measurements every five steps, C-PHAST reaches the lowest absolute Go2 error of any family, so the structural prior carries the open-loop regime while the learned typed dynamics carry tracking. Rollout evaluation makes this separation clearer than one-step prediction: several baselines fit the next state well, but degrade faster once their own predictions are fed back. The few-shot result points in the same direction: C-PHAST's zero-shot Go2 error remains lower than Shared-GRU after 50 target trajectories (Table 11). Together, these results suggest that storage, dissipation, and zero-power interconnection are useful invariants when the deployed body, graph, or observation interface changes.

C-PHAST also changes what a learned dynamics model gives to a decision layer. The Isaac physical-critic experiment shows that the model can flag unsafe candidate futures before simulator-level failure and rank supplied alternatives using typed physical factors (Table 9). The Spot-with-arm logs give the corresponding real-robot forecasting result: hardware telemetry and behavior descriptors are mapped to physically named future consequences and then to objective-conditioned action cards. In this view, the scalar score is not the model output. The reusable output is the typed consequence vector, which can be reweighted, vetoed, explained, or handed to a controller depending on the downstream objective. Section 5 gives the complementary controller-facing interface: because C-PHAST preserves power ports, an Energy–Casimir storage-shaping block can attach to the learned plant when the relevant matching and observation assumptions hold.

The non-robot benchmarks make the scope of the claim sharper. FHN supports repeated excitable-cell recomposition outside robotics, while the DC microgrid results show where typed recomposition and partial sensing matter in an electrical domain. The electrical evidence is intentionally nuanced: N-PHDAE remains a strong specialized raw forecaster in the full-state known-graph setting, while C-PHAST becomes most useful when the deployment contract changes through graph rewiring, typed block reuse, or voltage-only sensing. This is the intended role of the framework: a compositional world model for changing physical specifications, rather than a replacement for every fixed-domain pH solver. In this sense, C-PHAST operationalizes a narrow form of multiscale physical reasoning: for a declared typed decomposition, it forecasts how candidate futures route work, storage, dissipation, exchange, and residual mismatch across the modeled parts.

6.1. Scope and Outlook

The cleanest mathematical interface is the exact-port setting. The q-only experiments are important because they match common robot sensing, but they move part of the guarantee into the observer and canonicalizer: if the inferred port variables are poor, rollout quality and controller-side dissipation behavior can degrade. The Strang-split implementation preserves the intended structural decomposition, but large timesteps, stiffness, dense coupling, and sensor noise remain deployment issues rather than solved properties of the architecture. For larger systems, dense learned coupling can scale as $O(s^2)$ in the number of subsystems s, so sparse, hierarchical, and minimal-port constructions (Ehrhardt et al., 2026) are the natural scaling direction.

The structural power-preserving property (Theorem 3.1) is inherited from classical pH interconnection; the deployment-facing guarantees are the new theory. Section 3.8 and Appendices B–D establish them at three levels. Their constants are theorem hypotheses—analytically or numerically instantiated for the stated KNOWN-regime examples, and empirically calibrated for the trained and hybrid deployments—not universal upper bounds asserted for the learned stack: a finite-step energy-balance defect $H(x_{t+1}) - H(x_t) = W_t^{\text{port}} - D_t + \mathcal{O}(\Delta t^3)$ (Theorem B.1); an observer-aware deployment-transfer bound $e_h \le q^h \epsilon_{\text{lift}} + S_h(q)\bar{\delta}$ whose contraction factor q we verify in a Lyapunov metric rather than infer from passivity (Theorem C.2); and a typed-consequence decision-margin relation that certifies candidate rankings and constraints *whenever* the state-tube, readout, map-error, and constraint constants are certified on the operating region (Theorem D.2). Together

these put the transfer claim on the same footing as prior compositional-PHNN error analysis (Neary & Topcu, 2023) while extending it to the deployment setting. What remains is to certify the constants these theorems assume and to close their stated scope: proving—not only numerically verifying—the contraction factor for the trained nonlinear stack, supplying or probabilistically calibrating the readout map error ϵ^{map}, tightening the probabilistic transfer bound (Theorem C.4) for dependent, biased, or non-Gaussian observer refreshes and evaluating it in a genuinely measurement-closed rollout, and relaxing the matched-mode-schedule assumption to cover impacts and switching.

The main forward step is to close the loop around the same contract. In this paper, a measurement window refreshes the deployed state and then candidate futures are rolled forward. A deployed system should keep refreshing that state as new measurements arrive, while using typed consequences to guide candidate generation, vetoes, and controller interfaces. For Spot, this means promoting the current hardware interface blocks into a closed-loop rollout hierarchy; for microgrids and legged robots, it means using the same typed residuals to localize mismatch before adaptation. These are follow-up problems, but the present paper establishes the shared representation they would build on.

Acknowledgments

[To be added upon publication.]

References

Augner, B. Well-posedness and stability of infinite-dimensional linear port-Hamiltonian systems with nonlinear boundary feedback. *SIAM Journal on Control and Optimization*, 57(3):1818–1844, 2019.

Augner, B. Well-posedness and stability for interconnection structures of port-Hamiltonian type. In *Control Theory of Infinite-Dimensional Systems*, volume 277 of *Operator Theory: Advances and Applications*, pp. 1–52. Birkhäuser, 2020.

Bahety, A., Balaji, A., Abbatematteo, B., and Martín-Martín, R. SafeMimic: Towards safe and autonomous human-to-robot imitation for mobile manipulation. In *Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.128.

Bhardwaj, S. and Bajaj, C. PHAST: Port-hamiltonian architecture for structured temporal dynamics forecasting. *arXiv preprint arXiv:2602.17998*, 2026.

Bohlinger, N., Czechmanowski, G., Krupka, M. P., Kicki, P., Walas, K., Peters, J., and Tateo, D. One policy to run them all: An end-to-end learning approach to multi-embodiment locomotion. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3356–3378. PMLR, 2025.

Brantner, B., de Romemont, G., Kraus, M., and Li, Z. Volume-preserving transformers for learning time series data with structure. *arXiv preprint arXiv:2312.11166*, 2024.

Brogliato, B. *Nonsmooth Mechanics: Models, Dynamics and Control*. Springer, London, 1st edition, 1999.

Cebrián-Lacasa, D., Parra-Rivas, P., Ruiz-Reynés, D., and Gelens, L. Six decades of the fitzhugh-nagumo model: A guide through its spatio-temporal dynamics and influence across disciplines. *Physics Reports*, 1096:1–39, 2024. doi: 10.1016/j.physrep.2024.09.014.

Cervera, J., Van der Schaft, A., and Baños, A. Interconnection of port-Hamiltonian systems and composition of Dirac structures. *Automatica*, 43(2):212–225, 2007.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Cranmer, M., Greydanus, S., Hoyer, S., Battaglia, P., Spergel, D., and Ho, S. Lagrangian neural networks. *arXiv preprint arXiv:2003.04630*, 2020.

Cucuzzella, M., Trip, S., De Persis, C., Cheng, X., Ferrara, A., and van der Schaft, A. A robust consensus algorithm for current sharing and voltage regulation in DC microgrids. *IEEE Transactions on Control Systems Technology*, 27(4): 1583–1595, 2019.

Davis, C. and Kahan, W. M. The rotation of eigenvectors by a perturbation. iii. *SIAM Journal on Numerical Analysis*, 7(1): 1–46, 1970.

Desai, S. A., Mattheakis, M., Sondak, D., Protopapas, P., and Roberts, S. J. Port-hamiltonian neural networks for learning explicit time-dependent dynamical systems. *Physical Review E*, 104(3):034312, 2021.

Devin, C., Gupta, A., Darrell, T., Abbeel, P., and Levine, S. Learning modular neural network policies for multi-task and multi-robot transfer. In *IEEE International Conference on Robotics and Automation*, pp. 2169–2176, 2017.

Duong, T., Altawaitan, A., Stanley, J., and Atanasov, N. Port-Hamiltonian neural ODE networks on lie groups for robot dynamics learning and control. *IEEE Transactions on Robotics*, 2024.

Ehrhardt, M., Günther, M., and Ševčovič, D. Structure-preserving coupling and decoupling of port-Hamiltonian systems. *Applied Mathematics Letters*, 177:109894, 2026. doi: 10.1016/j.aml.2026.109894. URL https://doi.org/10.1016/j.aml.2026.109894.

FitzHugh, R. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical Journal*, 1(6): 445–466, 1961.

Furieri, L., Galimberti, C. L., Zakwan, M., and Ferrari-Trecate, G. Distributed neural network control with dependability guarantees: a compositional port-Hamiltonian approach. In *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, volume 168 of *Proceedings of Machine Learning Research*, pp. 571–583. PMLR, 2022. Preprint: arXiv:2112.09046.

Gernandt, H., Preuster, T., and Schaller, M. Optimization-based control by interconnection of nonlinear port-Hamiltonian systems. *arXiv preprint arXiv:2602.06670*, 2026.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pp. 1263–1272. PMLR, 2017.

Greydanus, S., Dzamba, M., and Yosinski, J. Hamiltonian neural networks. *Advances in neural information processing systems*, 32, 2019.

Ha, D. and Schmidhuber, J. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Hansen, N., Su, H., and Wang, X. TD-MPC2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2024.

Hasani, R., Wieser, E., Lechner, M., Kim, T. A., Amini, A., Rus, D., and Grosu, R. Linear oscillatory state space models. *arXiv preprint arXiv:2410.03943*, 2024.

Hu, J., Stone, P., and Martín-Martín, R. SLAC: Simulation-pretrained latent action space for whole-body real-world RL. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 2966–2982. PMLR, 2025.

Huang, W., Mordatch, I., and Pathak, D. One policy to control them all: Shared modular policies for agent-agnostic control. In *International Conference on Machine Learning*, pp. 4455–4464. PMLR, 2020.

Jin, P., Zhang, Z., Zhu, A., Tang, Y., and Karniadakis, G. E. Sympnets: Intrinsic structure-preserving symplectic networks for identifying hamiltonian systems. *Neural Networks*, 132:166–179, 2020.

Kumar, A., Fu, Z., Pathak, D., and Malik, J. RMA: Rapid motor adaptation for legged robots. In *Proceedings of Robotics: Science and Systems*, Virtual, 2021. doi: 10.15607/RSS.2021.XVII.011.

Kumar, A., Li, Z., Zeng, J., Pathak, D., Sreenath, K., and Malik, J. Adapting rapid motor adaptation for bipedal robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022. doi: 10.1109/IROS47612.2022.9981091.

Le Gorrec, Y., Zwart, H., and Maschke, B. Dirac structures and boundary control systems associated with skew-symmetric differential operators. *SIAM Journal on Control and Optimization*, 44(5):1864–1892, 2005.

Li, C., Krause, A., and Hutter, M. Robotic world model: A neural network simulator for robust policy optimization in robotics. *arXiv preprint arXiv:2501.10100*, 2025.

Li, P., Tan, K., Moyer, D., and Beckers, T. Identify then project: Contrastive learning of latent dynamics from partial observations with port-Hamiltonian structure. *arXiv preprint arXiv:2605.16682*, 2026.

Liu, Y., Meliopoulos, A. P. S., Sun, L., and Choi, S. Protection and control of microgrids using dynamic state estimation. *Protection and Control of Modern Power Systems*, 3(1):31, 2018.

Lutter, M., Ritter, C., and Peters, J. Deep Lagrangian networks: Using physics as model prior for deep learning. In *International Conference on Learning Representations*, 2019.

Maes, L., Le Lidec, Q., Facury, L., Massaudi, N., Chaurasia, A., Capuano, F., Gao, R., Gillin, T., Haramati, D., Scieur, D., LeCun, Y., and Balestriero, R. stable-worldmodel: A platform for reproducible world modeling research and evaluation, 2026. URL https://arxiv.org/abs/2605.21800.

Margolis, G. B. and Agrawal, P. Walk these ways: Tuning robot control for generalization with multiplicity of behavior. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 22–31. PMLR, 2023.

Moradi, S., Beintema, G. I., Jaensson, N., Tóth, R., and Schoukens, M. Port-Hamiltonian neural networks with output error noise models. *arXiv preprint arXiv:2502.14432*, 2025.

Neary, C. and Topcu, U. Compositional learning of dynamical system models using port-Hamiltonian neural networks. In Matni, N., Morari, M., and Pappas, G. J. (eds.), *Proceedings of The 5th Annual Learning for Dynamics and Control Conference*, volume 211 of *Proceedings of Machine Learning Research*, pp. 679–691. PMLR, 15–16 Jun 2023. URL https://proceedings.mlr.press/v211/neary23a.html.

Neary, C., Tsao, N., and Topcu, U. Neural port-Hamiltonian differential algebraic equations for compositional learning of electrical networks. *arXiv preprint arXiv:2412.11215*, 2024.

NVIDIA Isaac Lab Developers. Training a gear insertion policy and ros deployment. Isaac Lab Documentation, 2026. URL https://isaac-sim.github.io/IsaacLab/main/source/policy_deployment/02_gear_assembly/gear_assembly_policy.html. Accessed: 2026-04-14.

Ortega, R., Van Der Schaft, A., Maschke, B., and Escobar, G. Interconnection and damping assignment passivity-based control of port-controlled Hamiltonian systems. *Automatica*, 38(4):585–596, 2002.

Ortega, R., Van Der Schaft, A., Maschke, B., and Escobar, G. Control by interconnection and standard passivity-based control of port-hamiltonian systems. *IEEE Transactions on automatic control*, 53(11):2527–2542, 2008.

Orvieto, A., Smith, S. L., Gu, A., Fernando, A., Gulcehre, C., Pascanu, R., and De, S. Resurrecting recurrent neural networks for long sequences. In *International Conference on Machine Learning*, pp. 26670–26698. PMLR, 2023.

Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., and Battaglia, P. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2021.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378: 686–707, 2019.

Rudin, N., Hoeller, D., Reist, P., and Hutter, M. Learning to walk in minutes using massively parallel deep reinforcement learning. In *Conference on Robot Learning*, pp. 91–100. PMLR, 2022.

Sadabadi, M. S., Shafiee, Q., and Karimi, A. Plug-and-play robust voltage control of DC microgrids. *IEEE Transactions on Smart Grid*, 9(6):6886–6896, 2018.

Sanchez-Escalonilla, S., Zoboli, S., and Jayawardhana, B. Robust neural IDA-PBC: passivity-based stabilization under approximations. *arXiv preprint arXiv:2409.16008*, 2024.

Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P. W. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, pp. 8459–8468. PMLR, 2020.

Sleiman, J.-P., Farshidian, F., and Hutter, M. Versatile multicontact planning and control for legged loco-manipulation. *Science Robotics*, 8(81), 2023. doi: 10.1126/scirobotics.adg5014.

Smith, J. T., Warrington, A., and Linderman, S. W. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*, 2023.

Sosanya, A. and Greydanus, S. Dissipative hamiltonian neural networks: Learning dissipative and conservative dynamics separately. In *International Conference on Learning Representations Workshop on Physics for Machine Learning*, 2022.

Tropp, J. A. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4): 389–434, 2012.

Tropp, J. A. Applied random matrix theory. *arXiv preprint arXiv:2604.27119*, 2026.

Tucci, M., Riverso, S., and Ferrari-Trecate, G. Line-independent plug-and-play controllers for voltage stabilization in DC microgrids. *IEEE Transactions on Control Systems Technology*, 26(3):1115–1123, 2018. Preprint: arXiv:1609.02456.

Van Der Schaft, A. and Jeltsema, D. Port-Hamiltonian systems theory: An introductory overview. *Foundations and Trends in Systems and Control*, 1(2-3):173–378, 2014.

van Otterdijk, G. J. E., Moradi, S., Weiland, S., Tóth, R., Jaensson, N. O., and Schoukens, M. Learning subsystem dynamics in nonlinear systems via port-Hamiltonian neural networks. *arXiv preprint arXiv:2411.05730*, 2024.

Venegas-Aravena, P. and Cordaro, E. G. The multiscale principle in nature (principium luxuriæ): Linking multiscale thermodynamics to living and non-living complex systems. *Fractal and Fractional*, 8(1):35, 2024. doi: 10.3390/fractalfract8010035. URL https://www.mdpi.com/2504-3110/8/1/35.

Viernickel, J. H., Welschehold, T., Burgard, W., and Trimpe, S. Floating-base deep lagrangian networks. *arXiv preprint arXiv:2510.17270*, 2025.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R., and Smola, A. J. Deep sets. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Zakwan, M. and Ferrari-Trecate, G. Neural distributed controllers with port-Hamiltonian structures. In *2024 IEEE 63rd Conference on Decision and Control (CDC)*. IEEE, 2024. Preprint: arXiv:2403.17785.

Zhang, J. Z., Sorokin, M., Brüdigam, J., Hung, B., Phillips, S., Yershov, D., Niroui, F., Zhao, T., Fermoselle, L., Zhu, X., Cao, C., Ta, D., Pang, T., Wang, J., Culbertson, P., Manchester, Z., and Le Cléac'h, S. SUMO: Dynamic and generalizable whole-body loco-manipulation. *arXiv preprint arXiv:2604.08508*, 2026.

Zhong, Y. D., Dey, B., and Chakraborty, A. Dissipative SymODEN: Encoding Hamiltonian dynamics with dissipation and control into deep learning. *arXiv preprint arXiv:2002.08860*, 2020.

Zhou, G., Pan, H., LeCun, Y., and Pinto, L. DINO-WM: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024.

How to read this appendix. The appendix is organized by supporting claim rather than by artifact type. Appendices A and E first discharge the Section 5 control-compatibility claim: the first gives the exact-port stability calculation, and the second records the q-only implementation and compositional-control benchmark protocol. Appendix F then supports the Section 3 construction of the deployed predictor: canonicalization, PHASTCore, coupling, integration, and training. Appendix G consolidates the planar-arm material: protocol, schematic, companion table, coupling ablations, diagnostics, hyperparameters, and stale-URDF sanity checks. Appendix H mirrors the remaining benchmark order of Table 6: MuJoCo leg-count transfer, FitzHugh–Nagumo repeated excitable-cell transfer, Isaac Lab cross-robot transfer, and DC microgrid recomposition. Appendix I collects the legged-robot Isaac Lab diagnostics and target-data studies: jerk, observer ablation, few-shot Go2 adaptation, and target-data scaling. Appendix J collects the interface-facing robotics evidence: the real Spot shadow-mode information contract, the physical-critic interface, support-choice candidate ranking, and partial-flag damping-spectrum diagnostics.

A. Energy–Casimir Stability Analysis

This appendix provides the exact-port stability derivation for the Energy–Casimir controller introduced in Section 5. The q-only experiments later in Appendix E use this result as the ideal reference point and then evaluate how observer error changes the deployed behavior.

A.1. Setup

Consider a single-joint port-Hamiltonian system with state $x = (q, p)$, separable Hamiltonian $H(q, p) = V(q) + \frac{p^2}{2m}$, and damping $d_{\text{nat}} \geq 0$:

$$\dot{q} = \frac{\partial H}{\partial p} = \frac{p}{m}, \qquad \dot{p} = -\frac{\partial H}{\partial q} - d_{\text{nat}}\,\dot{q} + u. \tag{48}$$

The port output is $y = \frac{\partial H}{\partial p} = \dot{q}$ (velocity), and the passivity inequality gives:

$$\dot{H} = -d_{\text{nat}}\,\dot{q}^2 + u\,\dot{q} = -d_{\text{nat}}\,\dot{q}^2 + u\,y. \tag{49}$$

Thus the controller can change stored energy only through the port power $u\,y$; the rest of the argument tracks how the controller routes or dissipates that power.

A.2. Energy–Casimir Controller

To regulate the system to a target $q^\star$ (with zero velocity), we introduce a *Casimir/controller state* ξ that integrates the port output:

$$\dot{\xi} = \dot{q}, \qquad \xi(0) = q(0), \tag{50}$$

This implies the Casimir $\mathcal{C}(q, \xi) = q - \xi$ is invariant, so $\xi(t) = q(t)$ for all t — an achievable controller Casimir in the sense of Cervera et al. (2007). This lock is the mechanism by which a controller storage term in ξ becomes a shaped plant potential in q. The controller applies potential shaping plus damping injection:

$$u = -k_c(\xi - q^\star) - d_{\text{inj}}\,\dot{q}, \tag{51}$$

with shaping gain $k_c > 0$ and injected damping $d_{\text{inj}} > 0$. This is the scalar version of the main-text damping-injection matrix K_{inj}: for the arm controller, $K_{\text{inj}} = d_{\text{inj}} I$.

Physical interpretation. The term $-k_c(\xi - q^\star)$ adds a quadratic target preference, so on the Casimir manifold the closed-loop potential is $V(q) + \frac{k_c}{2}\|q - q^\star\|^2$. The stability result below requires this shaped potential to have an isolated local minimum at the desired target. The term $-d_{\text{inj}}\,\dot{q}$ adds dissipation beyond the natural damping d_{nat}, accelerating convergence. The Casimir state ξ locks to q through the port interconnection; in our discrete-time q-only implementation we additionally use drift correction to mitigate numerical drift (Remark A.2).

A.3. Lyapunov Analysis Under Exact Observation

Theorem A.1 (Closed-loop stability)**.** *Consider the port-Hamiltonian system* (48) *with the Energy–Casimir controller* (51) *and Casimir dynamics* (50)*. Assume exact velocity observation ($\hat{\dot{q}} = \dot{q}$), $k_c > 0$, $d_{\text{inj}} > 0$, and $d_{\text{nat}} \geq 0$, together with the*

standard local hypotheses $m > 0$, $V \in C^2$, local existence and uniqueness of solutions, and a compact positively invariant sublevel set of H_{shaped} contained in the stated neighborhood. Let $U_{\text{sh}}(q) = V(q) + \frac{k_c}{2}\|q - q^\star\|^2$. Assume that, in the local neighborhood considered, $q^\star$ is an isolated strict local minimum of U_{sh} and no other critical point of U_{sh} lies in that neighborhood. Since $U'_{\text{sh}}(q^\star) = V'(q^\star)$, this forces $V'(q^\star) = 0$: the shaping law adds stiffness and damping around a configuration that is already *an equilibrium of the open-loop potential, rather than relocating the equilibrium to an arbitrary target (which would require an added feedforward term $V'(q^\star)$). Then:*

1. *The shaped Hamiltonian*
$$H_{\text{shaped}}(q, p, \xi) = H(q, p) + \frac{k_c}{2}\|\xi - q^\star\|^2 \tag{52}$$
is a Lyapunov candidate and has a strict local minimum at $(q^\star, 0, q^\star)$ on the Casimir manifold.

2. *The time derivative satisfies*
$$\dot{H}_{\text{shaped}} = -(d_{\text{nat}} + d_{\text{inj}})\,\dot{q}^2 \le 0. \tag{53}$$

3. *Under LaSalle's invariance principle and the isolated-critical-point assumption above, the equilibrium $(q^\star, 0, q^\star)$ is locally asymptotically stable* relative to the initialized Casimir manifold $\mathcal{M}_0 = \{\xi = q\}$ *(the controller sets $\xi(0) = q(0)$). It is not asymptotically stable against perturbations that change the Casimir $q - \xi$: those move the state onto a different invariant manifold $\{q - \xi = c\}$.*

Proof. **Item 1: Lyapunov candidate.** With the separable $H(q, p) = V(q) + p^2/2m$, on the Casimir manifold $\xi = q$ the shaped Hamiltonian is $H_{\text{shaped}} = \frac{p^2}{2m} + U_{\text{sh}}(q)$: the kinetic energy, strictly minimized at $p = 0$, plus U_{sh}, a strict local minimum at $q^\star$ by assumption. It therefore has a strict local minimum at $(q^\star, 0, q^\star)$ on the manifold and is a valid Lyapunov candidate. Items 2–3 (the dissipation identity and asymptotic stability) are established next; Steps 1–4 give Item 2 and Step 5 gives Item 3.

Step 1: Time derivative of H. From the passivity inequality (49):
$$\dot{H} = -d_{\text{nat}}\,\dot{q}^2 + u\,\dot{q}. \tag{54}$$

Step 2: Substitute the controller. Inserting $u = -k_c(\xi - q^\star) - d_{\text{inj}}\,\dot{q}$:
$$\begin{aligned}\dot{H} &= -d_{\text{nat}}\,\dot{q}^2 + \big[-k_c(\xi - q^\star) - d_{\text{inj}}\,\dot{q}\big]\,\dot{q} \\ &= -(d_{\text{nat}} + d_{\text{inj}})\,\dot{q}^2 - k_c(\xi - q^\star)\,\dot{q}.\end{aligned} \tag{55}$$

Step 3: Time derivative of the Casimir term.
$$\begin{aligned}\frac{d}{dt}\frac{k_c}{2}\|\xi - q^\star\|^2 &= k_c(\xi - q^\star)\,\dot{\xi} \\ &= k_c(\xi - q^\star)\,\dot{q}.\end{aligned} \tag{56}$$

Step 4: Combine. Adding (55) and (56):
$$\begin{aligned}\dot{H}_{\text{shaped}} &= -(d_{\text{nat}} + d_{\text{inj}})\,\dot{q}^2 \underbrace{-k_c(\xi - q^\star)\dot{q} + k_c(\xi - q^\star)\dot{q}}_{\text{cancel}} \\ &= -(d_{\text{nat}} + d_{\text{inj}})\,\dot{q}^2.\end{aligned} \tag{57}$$

Step 5: LaSalle's invariance principle. Consider the set $\mathcal{S} = \{(q, p, \xi) : \dot{H}_{\text{shaped}} = 0\}$. From (57) we have $\dot{q} = 0$ on $\mathcal{S}$, hence $p = 0$ and $\dot{\xi} = \dot{q} = 0$. Because $\xi(0) = q(0)$ and $\xi = q$ is invariant, the dynamics reduce to the shaped equilibrium condition $0 = -\nabla V(q) - k_c(q - q^\star)$. By the isolated-critical-point assumption on U_{sh}, the only solution in the local neighborhood is $q = q^\star$. By LaSalle's principle, the largest invariant set in $\mathcal{S}$ is $\{(q^\star, 0, q^\star)\}$, giving asymptotic stability. □

Remark A.2 (Practical drift correction)**.** In discrete time, numerical drift and q-only measurement noise can violate the exact invariant $\xi = q$. We therefore use a drift-correction term (Section 5):

$$\dot{\xi} = \hat{\dot{q}} + k_\xi(\tilde{q} - \xi), \qquad k_\xi > 0, \tag{58}$$

which drives the measured position–controller mismatch $\tilde{q} - \xi$ toward zero with time constant $1/k_\xi$ when the velocity estimate is accurate. This correction introduces an additional term in the Lyapunov derivative, e.g., in the exact-position idealization $\dot{H}_{\text{shaped}} = -(d_{\text{nat}} + d_{\text{inj}})\dot{q}^2 + k_c k_\xi(\xi - q^\star)(q - \xi)$. This cross term is not sign-definite, so the drift correction should be read as an implementation mechanism rather than as part of the exact Lyapunov guarantee. When ξ tracks q, the additional term vanishes, which is why the correction improves robustness in long q-only rollouts.

A.4. Effect of Observer Error

Remark A.3 (Degradation under noisy velocity estimation)**.** In the q-only setting the controller uses $\hat{\dot{q}}$ instead of $\dot{q}$; let $e_t = \hat{\dot{q}}_t - \dot{q}_t$ be the observer error. We isolate the damping-injection channel here: *when the Casimir state is updated by the exact port output* ($\dot{\xi} = \dot{q}$), the estimate enters only the damping term $-d_{\text{inj}}(\dot{q} + e)$, and the Lyapunov derivative gains the single cross-term:

$$\dot{H}_{\text{shaped}} = -(d_{\text{nat}} + d_{\text{inj}})\,\dot{q}^2 - d_{\text{inj}}\,e\,\dot{q}. \tag{59}$$

By Young's inequality, $|d_{\text{inj}}\,e\,\dot{q}| \le \frac{d_{\text{inj}}}{2}\dot{q}^2 + \frac{d_{\text{inj}}}{2}e^2$, so:

$$\dot{H}_{\text{shaped}} \le -\Big(d_{\text{nat}} + \frac{d_{\text{inj}}}{2}\Big)\dot{q}^2 + \frac{d_{\text{inj}}}{2}e^2. \tag{60}$$

This bound should be read as a robustness calculation rather than a new stability theorem:

- **Observer error can act as energy injection**: the $\frac{d_{\text{inj}}}{2}e^2$ term is the price of using an estimated port velocity.
- **The damping margin is reduced in this bound**: the chosen Young-inequality split leaves the coefficient $(d_{\text{nat}} + d_{\text{inj}}/2)$ on $\dot{q}^2$.
- **Exact convergence requires controlled observer error**: if e is persistently large, the system may fail to converge; if e decays or remains small, the bound gives the expected practical robustness interpretation.

When the estimate *also* drives the Casimir update, $\dot{\xi} = \hat{\dot{q}} + k_\xi(\tilde{q} - \xi)$ (Remark A.2), the exact shaped-storage derivative—with velocity error $e_v = \hat{\dot{q}} - \dot{q}$ and position error $e_q = \tilde{q} - q$—is

$$\dot{H}_{\text{shaped}} = -(d_{\text{nat}} + d_{\text{inj}})\dot{q}^2 - d_{\text{inj}}e_v\dot{q} + k_c(\xi - q^\star)e_v + k_c k_\xi(\xi - q^\star)(q - \xi + e_q). \tag{61}$$

Beyond the damping channel $-d_{\text{inj}}e_v\dot{q}$ of (59), the estimate injects $k_c(\xi - q^\star)e_v$ (the velocity estimate enters the Casimir update) and the Casimir-lock term $k_c k_\xi(\xi - q^\star)(q - \xi + e_q)$; the first three terms alone describe only the exact-$\dot{\xi}$ idealization. A complete q-only robustness bound combines all channels and is a practical, not exact, guarantee. This analysis motivates the experimental comparison of velocity estimation strategies in Section 5 and Appendix E.

A.5. Compositional Stability Under Skew-Symmetric Coupling

The same calculation extends to the joint-level planar-arm setting, where scalar pH joint blocks are coupled through a skew-symmetric matrix $\hat{\mathbf{C}} = -\hat{\mathbf{C}}^\top$.

Theorem A.4 (Compositional closed-loop stability)**.** *Consider s scalar mechanical pH subsystems, each with state $x_i = (q_i, p_i)$, separable Hamiltonian $H_i(q_i, p_i) = V_i(q_i) + T_i(p_i)$ with strictly convex kinetic energy T_i—so that $\partial H_i/\partial p_i = 0 \iff p_i = 0$ (e.g. $T_i = p_i^2/2m_i$, $m_i > 0$), which is what lets $\dot{q}_i = 0$ imply $p_i = 0$—and damping $d_{\text{nat},i} \ge 0$, coupled via*

$$\dot{p}_i = -\frac{\partial H_i}{\partial q_i} - d_{\text{nat},i}\,\dot{q}_i + u_i + \hat{u}_i, \qquad \hat{u}_i = -\sum_{j=1}^{s}\hat{C}_{ij}\,\dot{q}_j, \tag{62}$$

where $\hat{\mathbf{C}} = -\hat{\mathbf{C}}^\top$ and $\hat{u}_i$ is the coupling-port input entering subsystem i. Apply the per-joint Energy–Casimir controller (51) independently to each joint:

$$u_i = -k_c(\xi_i - q_i^\star) - d_{\text{inj}}\,\dot{q}_i, \qquad \dot{\xi}_i = \dot{q}_i. \tag{63}$$

Here ξ_i is the controller-integrator state, $q_i^\star$ is the desired joint equilibrium, $k_c > 0$ is the shaping gain, and $d_{\mathrm{inj}} > 0$ is the injected damping. Under exact velocity observation, and assuming each shaped potential $U_{\mathrm{sh},i}(q_i) = V_i(q_i) + \frac{k_c}{2}\|q_i - q_i^\star\|^2$ has $q_i^\star$ as an isolated strict local minimum with no other local critical point in the neighborhood considered, the total shaped Hamiltonian

$$H_{\mathrm{shaped}}^{\mathrm{total}} = \sum_{i=1}^{s} \left[H_i(q_i, p_i) + \frac{k_c}{2}\|\xi_i - q_i^\star\|^2 \right] \tag{64}$$

satisfies

$$\dot{H}_{\mathrm{shaped}}^{\mathrm{total}} = -\sum_{i=1}^{s} (d_{\mathrm{nat},i} + d_{\mathrm{inj}})\, \dot{q}_i^2 \le 0. \tag{65}$$

In particular, the coupling term $\hat{u}$ contributes zero *net power and does not appear in $\dot{H}_{\mathrm{shaped}}^{\mathrm{total}}$. Consequently, the stacked equilibrium $(q^\star, 0, q^\star)$ is locally asymptotically stable—again relative to the initialized manifold $\mathcal{M}_0 = \{\xi_i = q_i\}$, and requiring each $q_i^\star$ to satisfy $V_i'(q_i^\star) = 0$—under the same local LaSalle argument as in Theorem A.1.*

Proof. **Step 1: Coupling power vanishes.** Let $\dot{q} = [\dot{q}_1, \ldots, \dot{q}_s]^\top$ collect the subsystem velocities. The total coupling power is:

$$\sum_{i=1}^{s} \dot{q}_i\, \hat{u}_i = -\sum_{i=1}^{s}\sum_{j=1}^{s} \dot{q}_i\, \hat{C}_{ij}\, \dot{q}_j = -\dot{q}^\top \hat{\mathbf{C}}\, \dot{q} = 0, \tag{66}$$

since $\hat{\mathbf{C}} = -\hat{\mathbf{C}}^\top$ implies $v^\top \hat{\mathbf{C}}\, v = 0$ for all v (the associated quadratic form of a skew-symmetric matrix vanishes identically).

Step 2: Sum per-joint Lyapunov derivatives. From the single-joint analysis (57), each joint's shaped Hamiltonian satisfies:

$$\dot{H}_{\mathrm{shaped},i} = -(d_{\mathrm{nat},i} + d_{\mathrm{inj}})\, \dot{q}_i^2 + \dot{q}_i\, \hat{u}_i, \tag{67}$$

where the $\dot{q}_i\, \hat{u}_i$ term accounts for the coupling port power entering joint i. Summing over all joints:

$$\begin{aligned} \dot{H}_{\mathrm{shaped}}^{\mathrm{total}} &= \sum_{i=1}^{s} \dot{H}_{\mathrm{shaped},i} = -\sum_{i=1}^{s} (d_{\mathrm{nat},i} + d_{\mathrm{inj}})\, \dot{q}_i^2 + \underbrace{\sum_{i=1}^{s} \dot{q}_i\, \hat{u}_i}_{=\,0 \text{ by } (66)} \\ &= -\sum_{i=1}^{s} (d_{\mathrm{nat},i} + d_{\mathrm{inj}})\, \dot{q}_i^2 \;\le\; 0. \end{aligned} \tag{68}$$

Step 3: LaSalle's principle. The LaSalle argument from Theorem A.1 applies to the stacked system $(q, p, \xi) \in \mathbb{R}^{3s}$ with the stated local shaped-potential assumptions, since the coupling vanishes on the invariant set $\{\dot{q} = 0\}$. □

Remark A.5 (Exact-port controller reuse under repeated local assumptions)**.** Theorem A.4 has a practical consequence for C-PHAST: because the per-joint controller (63) uses shared parameters (k_c, d_{inj}) and the coupling cancels from the Lyapunov analysis, the *same* controller can be deployed on any number of joints $s' \neq s$ without retuning, provided every added joint satisfies the same local shaped-potential and kinetic assumptions and the trajectory stays in the local basin. The statement is local and conditional—it does not assert a count-uniform basin, convergence rate, or transient guarantee. It reuses the shared-block mechanism of the dynamics-prediction count transfer, but only in the exact-port idealization and as a local, conditional statement—not a control-theoretic transfer guarantee.

B. Finite-Step Passivity Defect of the Structure-Preserving Rollout

The continuous-time interconnection satisfies the exact power balance $\dot{H} = y^\top u - e^\top R\, e$ with $e = \nabla H$ (Theorem 3.1 for the internal coupling; the external ports and dissipation supply the remaining terms). This appendix quantifies how much of that balance survives one *finite* structure-preserving step, closing the caveat that continuous-time passivity does not automatically become a finite-step, observed-coordinate guarantee. The rearranged residual $H(x_{t+1}) - H(x_t) - (W_t^{\mathrm{port}} - D_t)$—i.e. $\Delta H +$ dissipation $-$ port work, which by the theorem below is the $\mathcal{O}(\Delta t^3)$ defect—is the finite-step energy-balance residual, and it is what the energy ledger accounts each step, not $H(x_{t+1}) - H(x_t)$ alone.

Theorem B.1 (Finite-step energy-balance defect)**.** *Let one canonical transition $x_t \mapsto x_{t+1}$ be produced by the second-order structure-preserving integrator (Strang splitting (18) of the leapfrog conservative step and the metric-preserving Cayley coupling with the damping half-step (19)) at timestep Δt, and account the boundary power terms by the trapezoidal rule (under the zero-order hold below the endpoint input is u_t at both nodes)*

$$W_t^{\mathrm{port}} = \tfrac{\Delta t}{2}\big(y_t^\top u_t + y_{t+1}^\top u_t\big), \qquad D_t = \tfrac{\Delta t}{2}\big(e_t^\top R\, e_t + e_{t+1}^\top R\, e_{t+1}\big) \geq 0. \tag{69}$$

Assume $H \in C^3$ with bounded derivatives on a forward-invariant set containing the trajectory, that $s \mapsto P(x(s), u)$ has a bounded second derivative over the step (a direct hypothesis on the boundary power $P(x,u) = y^\top u - e^\top Re$, not inferred from Lipschitzness of the field alone), a zero-order-held input on $[t, t+\Delta t]$, and no mode/contact switch within the step. Assume moreover that the integrator is second order on the step; this holds for the leapfrog core with the damping half-step (19) in the constant identity, scalar, or blockwise-scalar mass regime, and excludes a configuration-dependent $M(q)$, a configuration-dependent coupling $\hat{\mathbf{C}}(q)$ (whose opening and closing coupling half-steps would otherwise use different generators $A(q_t) \neq A(q_{t+1})$ and break the second-order symmetry unless A is evaluated at both endpoints), or a start-of-step-only state-dependent domain port. Then

$$H(x_{t+1}) - H(x_t) = W_t^{\mathrm{port}} - D_t + \mathcal{O}(\Delta t^3). \tag{70}$$

Under q-only deployment, where the endpoints are replaced by observer/chart estimates $\hat{x}_t, \hat{x}_{t+1}$ with state errors $\eta_s = \|\hat{x}_s - x_s\| \leq \epsilon_{\mathrm{obs}} + \epsilon_{\mathrm{chart}}$, the estimated-state defect obeys

$$\big|H(\hat{x}_{t+1}) - H(\hat{x}_t) - (\widehat{W}_t^{\mathrm{port}} - \widehat{D}_t)\big| \;\leq\; C\,\Delta t^3 + \big(L_H + \tfrac{\Delta t}{2} L_P\big)\,(\eta_t + \eta_{t+1}), \tag{71}$$

where $\widehat{W}_t^{\mathrm{port}}, \widehat{D}_t$ are the trapezoidal terms evaluated at the estimates, L_H is a local Lipschitz constant of H, and L_P one of the boundary power P; the $\frac{\Delta t}{2} L_P$ contribution comes from perturbing the boundary-power quadrature and is absent from a naive L_H-only bound. If instead the boundary terms are accounted by a first-order (left-endpoint) rule, the $\mathcal{O}(\Delta t^3)$ remainder in (70) degrades to $\mathcal{O}(\Delta t^2)$; the third-order remainder therefore requires the second-order boundary quadrature together with *the stated smoothness, zero-order-hold, and no-switch conditions (these are sufficient, not merely necessary).*

Proof. Recall the exact power balance $\dot{H} = P(x,u)$ with continuous port power $P(x,u) = y^\top u - e^\top Re$ and effort $e = \nabla H$; the trapezoidal accounting of the theorem is $W_t^{\mathrm{port}} - D_t = \frac{\Delta t}{2}\big(P(x_t,u_t) + P(x_{t+1},u_t)\big)$ under the zero-order hold. We prove (70) by writing its defect as a quadrature error plus a state-truncation error, each $\mathcal{O}(\Delta t^3)$; the left-endpoint degradation and the q-only bound then follow.

Step 1 (exact balance). Under the zero-order hold u is constant on $[t, t+\Delta t]$, and integrating $\dot{H} = P$ along the exact flow $x(\cdot)$ from $x(t) = x_t$ gives

$$H\big(x(t{+}\Delta t)\big) - H(x_t) = \int_t^{t+\Delta t} P\big(x(s), u\big)\, ds.$$

Step 2 (quadrature error). By hypothesis $s \mapsto P(x(s),u)$ has a bounded second derivative over the step, so the trapezoidal rule applied to the Step-1 integral has local error $\mathcal{O}(\Delta t^3)$:

$$\int_t^{t+\Delta t} P\, ds = \big(W_t^{\mathrm{port}} - D_t\big) + \mathcal{O}(\Delta t^3).$$

(The trapezoidal terminal node is the numerical x_{t+1}, which the ledger uses; replacing it by the exact $x(t{+}\Delta t)$ shifts the terminal power by $\frac{\Delta t}{2} L_P \cdot \mathcal{O}(\Delta t^3) = \mathcal{O}(\Delta t^4)$, absorbed in the remainder.)

Step 3 (state-truncation error). The numerical endpoint x_{t+1} differs from the exact flow $x(t{+}\Delta t)$ only by the integrator's local truncation error, which is $\mathcal{O}(\Delta t^3)$ because the step is second-order and time-symmetric (Strang splitting of the conservative leapfrog step with the dissipative half-steps). Since ∇H is bounded and Lipschitz on the forward-invariant set,

$$H(x_{t+1}) - H\big(x(t{+}\Delta t)\big) = \nabla H \cdot \mathcal{O}(\Delta t^3) = \mathcal{O}(\Delta t^3).$$

The Cayley conservative variant obeys the same bound: for a G-skew generator ($A^\top G + GA = 0$) the Cayley transform is G-orthogonal, so it preserves $\|\cdot\|_G$ exactly while approximating $e^{\Delta t A}$ to $\mathcal{O}(\Delta t^3)$, leaving its local truncation error $\mathcal{O}(\Delta t^3)$.

Table 17. **Finite-step defect scaling (measured).** Log–log slope of the one-step energy-balance residual versus Δt on an untrained harmonic-oscillator pH block, double precision. Slopes match Theorem B.1. The trapezoidal driven slope is the resolved-regime value ($\Delta t \in [0.02, 0.16]$; below Δt=0.02 the residual reaches the $\sim 10^{-9}$ numerical floor).

System	Boundary-power rule	Measured slope	Theorem
conservative (u=0, R=0)	— (no boundary terms)	2.99	$\mathcal{O}(\Delta t^3)$
driven + damped	left-endpoint (rectangle)	1.98	$\mathcal{O}(\Delta t^2)$
driven + damped	trapezoidal (implementation)	$\approx$3.0–3.4	$\mathcal{O}(\Delta t^3)$

Step 4 (combine). Adding and subtracting $H(x(t{+}\Delta t))$ and using Step 1 splits the defect into exactly the two errors just bounded:

$$H(x_{t+1}) - H(x_t) - \big(W_t^{\mathrm{port}} - D_t\big) = \underbrace{H(x_{t+1}) - H\big(x(t{+}\Delta t)\big)}_{\mathcal{O}(\Delta t^3),\ \text{Step 3}} + \underbrace{\int_t^{t+\Delta t} P\,ds - \big(W_t^{\mathrm{port}} - D_t\big)}_{\mathcal{O}(\Delta t^3),\ \text{Step 2}}.$$

Both terms are $\mathcal{O}(\Delta t^3)$, which proves (70).

Step 5 (left-endpoint rule). With the first-order left-endpoint rule only Step 2 changes: its local error becomes $\mathcal{O}(\Delta t^2)$, so the quadrature term above is $\mathcal{O}(\Delta t^2)$ and the remainder degrades to $\mathcal{O}(\Delta t^2)$. The state-truncation term is unaffected; hence the third-order remainder needs the second-order boundary quadrature *together with* the stated smoothness, zero-order-hold, and no-switch conditions.

Step 6 (q-only deployment). Replacing the endpoints by estimates $\hat{x}_t, \hat{x}_{t+1}$ with errors $\eta_s = \|\hat{x}_s - x_s\|$ perturbs the defect of Step 4 in two Lipschitz-controlled places: the energy difference $H(\hat{x}_{t+1}) - H(\hat{x}_t)$ changes by at most $L_H(\eta_t + \eta_{t+1})$, and the trapezoidal term $\widehat{W}_t^{\mathrm{port}} - \widehat{D}_t = \frac{\Delta t}{2}\big(P(\hat{x}_t, u_t) + P(\hat{x}_{t+1}, u_t)\big)$ changes by at most $\frac{\Delta t}{2} L_P(\eta_t + \eta_{t+1})$. Writing the $\mathcal{O}(\Delta t^3)$ of Step 4 as $C\Delta t^3$ and adding these two perturbations gives the stated estimated-state bound; the $\frac{\Delta t}{2} L_P$ term is precisely the boundary-quadrature perturbation that an L_H-only bound omits. □

Empirical validation. We verify Theorem B.1 directly on the implementation, evaluating the finite-step energy-balance residual on an analytically specified, *untrained* (KNOWN-regime) system—a harmonic-oscillator pH block—advanced one step over a geometric Δt sweep in double precision (single precision reaches its rounding floor before the scaling is resolved). Table 17 reports the measured log–log slope of the one-step defect versus Δt. The conservative case isolates the symplectic remainder ($\mathcal{O}(\Delta t^3)$); the driven, damped case exercises the boundary-power quadrature, and the measured slope confirms both that a left-endpoint rule yields $\mathcal{O}(\Delta t^2)$ and that the trapezoidal accounting of (70) restores $\mathcal{O}(\Delta t^3)$. The implementation uses the trapezoidal rule, so the logged residual is a genuine $\mathcal{O}(\Delta t^3)$ energy-balance defect. Because this defect is sign-indefinite, it yields a discrete passivity inequality $H(x_{t+1}) - H(x_t) \le W_t^{\mathrm{port}}$ only on intervals where the dissipated energy exceeds it, $D_t \ge C\Delta t^3$ (or, under estimated-state accounting, $D_t \ge C\Delta t^3 + (L_H + \frac{\Delta t}{2} L_P)(\eta_t + \eta_{t+1})$). We draw no unconditional discrete-passivity conclusion for conservative steps or near zero dissipation—in a damping nullspace, or when the velocity scales with Δt—where the defect can dominate.

C. Observer-Aware Deployment-Transfer Bound

Section 3.8 states the horizon bound (37); here we give the hypotheses and proof. It is a comparison-map contraction estimate: it separates decay of the initial observer error from accumulation of the per-step model defect, and its usefulness rests on a contraction factor that is *measured*, not inferred from passivity.

Setup. Fix a deployment $\mathcal{S}$. Let $F_k : \mathcal{X}_\mathcal{S} \to \mathcal{X}_\mathcal{S}$ be the true one-step flow over $[t_k, t_{k+1}]$ (step Δt) under a fixed input (u_k) and mode schedule, $\widehat{F}_k : \mathcal{Z}_\mathcal{S} \to \mathcal{Z}_\mathcal{S}$ the deployed structure-preserving rollout, and $\Pi_\mathcal{S} : \mathcal{X}_\mathcal{S} \to \mathcal{Z}_\mathcal{S}$ an alignment map (physical and learned states may use different charts, coordinates, or dimensions). Let $\Omega_\mathcal{S} \subset \mathcal{Z}_\mathcal{S}$ be a forward-invariant operating region and M a *fixed* symmetric positive-definite metric on $\Omega_\mathcal{S}$ inducing the norm $\|v\|_M = (v^\top M v)^{1/2}$— a Lyapunov metric for the deployed map, which for nonlinear systems is a region-local choice constructed as in the discrete-Lyapunov solve below. Write $e_k = \|\Pi_\mathcal{S} x_k - \hat{z}_k\|_M$ and $\Omega_\mathcal{S}^X = \Pi_\mathcal{S}^{-1}(\Omega_\mathcal{S})$.

The commutation defect δ_k in hypothesis (A3) below is *derived*, not assumed: the following lemma bounds it from a pointwise transported vector-field mismatch together with the state-map consistency (A2), by a Grönwall estimate in a fixed

RMS-normalized metric. This makes (A2) load-bearing, states the interconnection through a generic operator $K(\mathcal{S})$ (the microgrid incidence norm being a corollary), and yields deployment-size-uniform constants. Theorem C.2 instantiates it at a single deployment, writing $M = M_{\mathcal{S}}$ and dropping the $\mathcal{S}$ subscripts.

Lemma C.1 (One-step commutation defect)**.** *Fix a deployment $\mathcal{S}$, a step size $\Delta t > 0$, and an admissible input $u_k \in \mathcal{U}_{\mathcal{S}}$ held fixed on the step. Let $f_{\mathcal{S}}$ and $\widehat{f}_{\mathcal{S}}$ generate unique flows on $\mathcal{X}_{\mathcal{S}}$ and $\mathcal{Z}_{\mathcal{S}}$, respectively. Write $\widehat{\varphi}_t(z;u) = \varphi_t^{\widehat{f}_{\mathcal{S}}}(z;u)$ and set*

$$F_k(x) = \varphi_{\Delta t}^{f_{\mathcal{S}}}(x;u_k), \qquad \widehat{F}_k(z) = \Psi_{\Delta t}(z,u_k).$$

Let $\Pi_{\mathcal{S}} : \mathcal{X}_{\mathcal{S}} \to \mathcal{Z}_{\mathcal{S}}$ be C^1, let $\Omega_{\mathcal{S}} \subset \mathcal{Z}_{\mathcal{S}}$, and write $\Omega_{\mathcal{S}}^X = \Pi_{\mathcal{S}}^{-1}(\Omega_{\mathcal{S}})$. For this deployment let $M_{\mathcal{S}} \succ 0$ be a constant (state-independent) metric and let

$$\|v\|_{M_{\mathcal{S}}} = (v^\top M_{\mathcal{S}} v)^{1/2}, \qquad \langle v,w\rangle_{M_{\mathcal{S}}} = v^\top M_{\mathcal{S}} w.$$

Assume the following.

(H1) State-map consistency (A2). *For some $K_{\Psi,\mathcal{S}} \geq 0$,*

$$\sup_{z\in\Omega_{\mathcal{S}}} \left\|\widehat{\varphi}_{\Delta t}(z;u_k) - \Psi_{\Delta t}(z,u_k)\right\|_{M_{\mathcal{S}}} \leq K_{\Psi,\mathcal{S}}\Delta t^3.$$

(H2) Pointwise transported-field mismatch. *For $x \in \Omega_{\mathcal{S}}^X$ define, without requiring an inverse of $\Pi_{\mathcal{S}}$,*

$$d_{\mathcal{S}}(x,u) := D\Pi_{\mathcal{S}}(x) f_{\mathcal{S}}(x,u) - \widehat{f}_{\mathcal{S}}(\Pi_{\mathcal{S}}x,u). \tag{72}$$

Suppose there are defect fields r_{loc}, r_{edge}, and r_{port}, an edge-residual metric $N_{\mathcal{S}} \succ 0$, and a linear deployed interconnection operator $K(\mathcal{S})$ such that

$$d_{\mathcal{S}} = r_{\mathrm{loc}} + K(\mathcal{S}) r_{\mathrm{edge}} + r_{\mathrm{port}},$$

and nonnegative constants such that, uniformly over the stated operating region and admissible inputs,

$$\|r_{\mathrm{loc}}\|_{M_{\mathcal{S}}} \leq \delta_{\mathrm{loc},\mathcal{S}}, \qquad \|r_{\mathrm{edge}}\|_{N_{\mathcal{S}}} \leq \delta_{\mathrm{edge},\mathcal{S}}, \qquad \|r_{\mathrm{port}}\|_{M_{\mathcal{S}}} \leq \delta_{\mathrm{port},\mathcal{S}}.$$

Thus, with the induced operator norm,

$$\sup_{x\in\Omega_{\mathcal{S}}^X,\, u\in\mathcal{U}_{\mathcal{S}}} \|d_{\mathcal{S}}(x,u)\|_{M_{\mathcal{S}}} \leq \delta_{\mathrm{vf},\mathcal{S}} := \delta_{\mathrm{loc},\mathcal{S}} + \|K(\mathcal{S})\|_{N_{\mathcal{S}}\to M_{\mathcal{S}}}\delta_{\mathrm{edge},\mathcal{S}} + \delta_{\mathrm{port},\mathcal{S}}. \tag{73}$$

If $\Pi_{\mathcal{S}}$ is a C^1 diffeomorphism onto its image, (72) is equivalently the Z-space field $D\Pi_{\mathcal{S}}(\Pi_{\mathcal{S}}^{-1}z) f_{\mathcal{S}}(\Pi_{\mathcal{S}}^{-1}z,u) - \widehat{f}_{\mathcal{S}}(z,u)$ used in the main-text notation.

(H3) Incremental bound. *There is a nonnegative $L_{f,\mathcal{S}}$ such that, for all relevant z, z' and $u \in \mathcal{U}_{\mathcal{S}}$,*

$$\left\langle z - z', \widehat{f}_{\mathcal{S}}(z,u) - \widehat{f}_{\mathcal{S}}(z',u)\right\rangle_{M_{\mathcal{S}}} \leq L_{f,\mathcal{S}}\|z-z'\|_{M_{\mathcal{S}}}^2. \tag{74}$$

An $M_{\mathcal{S}}$-Lipschitz bound with constant $L_{f,\mathcal{S}}$ is sufficient. When the field is differentiable, a uniform logarithmic-norm bound on a convex neighborhood containing the relevant line segments, $\mu_{M_{\mathcal{S}}}(D\widehat{f}_{\mathcal{S}}) \leq L_{f,\mathcal{S}}$, is also sufficient. A negative available upper bound may be relaxed to $L_{f,\mathcal{S}} = 0$ for the estimate below.

(H4) Within-step invariance. *For every $x_0 \in \Omega_{\mathcal{S}}^X$, both $\Pi_{\mathcal{S}}\varphi_t^{f_{\mathcal{S}}}(x_0;u_k)$ and $\widehat{\varphi}_t(\Pi_{\mathcal{S}}x_0;u_k)$ remain in $\Omega_{\mathcal{S}}$ for $0 \leq t \leq \Delta t$. In particular, there is no unmatched mode or contact event within the step.*

Define

$$\alpha(L,\Delta t) = \begin{cases} (e^{L\Delta t}-1)/L, & L > 0, \\ \Delta t, & L = 0. \end{cases}$$

Then the one-step commutation defect satisfies

$$\begin{aligned}\delta_k &:= \sup_{x\in\Omega_{\mathcal{S}}^X} \left\|\Pi_{\mathcal{S}} F_k(x) - \widehat{F}_k(\Pi_{\mathcal{S}} x)\right\|_{M_{\mathcal{S}}} \\ &\le \delta_{\mathrm{vf},\mathcal{S}}\alpha(L_{f,\mathcal{S}},\Delta t) + K_{\Psi,\mathcal{S}}\Delta t^3 \\ &\le \Delta t e^{L_{f,\mathcal{S}}\Delta t}\left(\delta_{\mathrm{loc},\mathcal{S}} + \|K(\mathcal{S})\|_{N_{\mathcal{S}}\to M_{\mathcal{S}}}\delta_{\mathrm{edge},\mathcal{S}} + \delta_{\mathrm{port},\mathcal{S}}\right) + K_{\Psi,\mathcal{S}}\Delta t^3. \end{aligned} \tag{75}$$

In particular, $\Delta t e^{L_{f,\mathcal{S}}\Delta t} = \Delta t(1+\mathcal{O}(\Delta t))$ *for fixed* $L_{f,\mathcal{S}}$.

For a family of deployments with $b_{\mathcal{S}}$ *explicit blocks, take the RMS metric*

$$M_{\mathcal{S}} = \frac{1}{b_{\mathcal{S}}}\,\mathrm{blockdiag}(M_{\mathcal{S},1},\dots,M_{\mathcal{S},b_{\mathcal{S}}}), \qquad 0 < m_- I \preceq M_{\mathcal{S},i} \preceq m_+ I, \tag{76}$$

with m_-, m_+ *independent of* $\mathcal{S}$*, and use analogous per-edge and per-port RMS metrics. If the block, edge, and port RMS defects are bounded uniformly by* $\bar{\delta}_{\mathrm{loc}}$*,* $\bar{\delta}_{\mathrm{edge}}$*, and* $\bar{\delta}_{\mathrm{port}}$*, and if*

$$\sup_{\mathcal{S}} L_{f,\mathcal{S}} \le \bar{L}_f, \quad \sup_{\mathcal{S}} K_{\Psi,\mathcal{S}} \le \bar{K}_\Psi, \quad \sup_{\mathcal{S}} \|K(\mathcal{S})\|_{N_{\mathcal{S}}\to M_{\mathcal{S}}} \le \bar{K}, \tag{77}$$

then

$$\sup_{\mathcal{S}} \delta_k \le \Delta t e^{\bar{L}_f\Delta t}\left(\bar{\delta}_{\mathrm{loc}} + \bar{K}\bar{\delta}_{\mathrm{edge}} + \bar{\delta}_{\mathrm{port}}\right) + \bar{K}_\Psi \Delta t^3, \tag{78}$$

which is independent of $b_{\mathcal{S}}$ *and of the number of edges or ports.*

Proof. The defect δ_k compares the exact projected flow $\Pi_{\mathcal{S}} F_k$ with the numerical map $\widehat{F}_k = \Psi_{\Delta t}$. We split it into a *flow-versus-flow* error, controlled by the vector-field mismatch through a one-sided Grönwall estimate, and a *flow-versus-map* error, controlled directly by the state-map consistency (H1); the RMS metric then removes the dependence on the block count.

Step 1 (transported dynamics). Fix $x_0 \in \Omega_{\mathcal{S}}^X$ and write

$$x(t) = \varphi_t^{f_{\mathcal{S}}}(x_0;u_k), \qquad w(t) = \Pi_{\mathcal{S}} x(t), \qquad \widehat{z}(t) = \widehat{\varphi}_t(\Pi_{\mathcal{S}} x_0;u_k)$$

for the exact trajectory, its projection, and the reduced flow from the same projected start. The chain rule and the transported-defect definition (72) give

$$\dot{w}(t) = D\Pi_{\mathcal{S}}(x(t)) f_{\mathcal{S}}(x(t),u_k) = \widehat{f}_{\mathcal{S}}(w(t),u_k) + d_{\mathcal{S}}(x(t),u_k),$$

so the projected exact flow follows the reduced field *plus* the mismatch $d_{\mathcal{S}}$.

Step 2 (Grönwall on the flow-versus-flow error). Let $e(t) = w(t) - \widehat{z}(t)$; then $e(0) = 0$ and

$$\dot{e}(t) = \widehat{f}_{\mathcal{S}}(w(t),u_k) - \widehat{f}_{\mathcal{S}}(\widehat{z}(t),u_k) + d_{\mathcal{S}}(x(t),u_k).$$

Differentiating the metric norm and applying the incremental bound (H3) to the field difference and the vector-field bound (73) of (H2) to the mismatch—both valid along the step because (H4) keeps $w(t)$ and $\widehat{z}(t)$ in $\Omega_{\mathcal{S}}$—gives

$$\begin{aligned}\frac{1}{2}\frac{d}{dt}\|e(t)\|_{M_{\mathcal{S}}}^2 &= \left\langle e(t), \widehat{f}_{\mathcal{S}}(w(t),u_k) - \widehat{f}_{\mathcal{S}}(\widehat{z}(t),u_k)\right\rangle_{M_{\mathcal{S}}} + \langle e(t), d_{\mathcal{S}}(x(t),u_k)\rangle_{M_{\mathcal{S}}} \\ &\le L_{f,\mathcal{S}}\|e(t)\|_{M_{\mathcal{S}}}^2 + \delta_{\mathrm{vf},\mathcal{S}}\|e(t)\|_{M_{\mathcal{S}}}.\end{aligned}$$

Writing $r(t) = \|e(t)\|_{M_{\mathcal{S}}}$ and using the upper Dini derivative D^+ where $r = 0$, this is

$$D^+ r(t) \le L_{f,\mathcal{S}} r(t) + \delta_{\mathrm{vf},\mathcal{S}}, \qquad r(0) = 0,$$

and Grönwall's comparison inequality yields

$$\|e(\Delta t)\|_{M_{\mathcal{S}}} \le \delta_{\mathrm{vf},\mathcal{S}}\alpha(L_{f,\mathcal{S}},\Delta t). \tag{79}$$

Step 3 (add the flow-versus-map error via A2). The exact projected flow and the numerical map differ only through the reduced flow $\widehat{\varphi}_{\Delta t}$ at the endpoint, so

$$\begin{aligned}&\Pi_{\mathcal{S}} F_k(x_0) - \Psi_{\Delta t}(\Pi_{\mathcal{S}} x_0, u_k)\\ &\quad= \underbrace{\Pi_{\mathcal{S}} F_k(x_0) - \widehat{\varphi}_{\Delta t}(\Pi_{\mathcal{S}} x_0; u_k)}_{e(\Delta t),\ \text{Step 2}} + \underbrace{\widehat{\varphi}_{\Delta t}(\Pi_{\mathcal{S}} x_0; u_k) - \Psi_{\Delta t}(\Pi_{\mathcal{S}} x_0, u_k)}_{\le K_{\Psi,\mathcal{S}} \Delta t^3 \ \text{by (H1)}}.\end{aligned}$$

The first bracket is $e(\Delta t)$, bounded by (79); the second is exactly the state-map consistency of (H1)/A2, at most $K_{\Psi,\mathcal{S}} \Delta t^3$. Taking norms and the supremum over x_0 gives the first inequality of (75), and the envelope follows from

$$\alpha(L, \Delta t) = \int_0^{\Delta t} e^{L(\Delta t - s)}\, ds \le \Delta t e^{L \Delta t} \qquad (L \ge 0).$$

Step 4 (size-uniformity via the RMS metric). For the RMS metric (76), a stacked defect $r = (r_1, \dots, r_{b_{\mathcal{S}}})$ has

$$\|r\|_{M_{\mathcal{S}}}^2 = \frac{1}{b_{\mathcal{S}}} \sum_{i=1}^{b_{\mathcal{S}}} \|r_i\|_{M_{\mathcal{S},i}}^2,$$

so a uniform per-block bound is an RMS bound of the *same* magnitude, not a $\sqrt{b_{\mathcal{S}}}$ multiple; the edge and port channels are identical. Substituting the uniform bounds (77) into (75) gives the size-independent (78). □

Incidence corollary (edge-explicit node–edge composition). Let $B = B_{\mathrm{inc}}(\mathcal{S}) \in \mathbb{R}^{n \times m}$ be an oriented incidence matrix. In compatible whitened port coordinates, the two off-diagonal mismatch channels are routed by

$$\mathcal{K}_B \begin{bmatrix} r_N \\ r_E \end{bmatrix} = \begin{bmatrix} B r_E \\ -B^\top r_N \end{bmatrix}, \qquad \mathcal{K}_B = \begin{bmatrix} 0 & B \\ -B^\top & 0 \end{bmatrix}.$$

With the total-block RMS metric $(n+m)^{-1} I$ (or its compatible block version), the scalar normalization cancels in the induced norm and

$$\|\mathcal{K}_B\|_M^2 = \lambda_{\max}(\mathcal{K}_B^\top \mathcal{K}_B) = \max\{\lambda_{\max}(BB^\top), \lambda_{\max}(B^\top B)\} = \|B\|_2^2. \tag{80}$$

The same identity holds for $B \otimes I_d$; here the concatenated r_N, r_E vector is the RMS edge-channel defect appearing in (H2). If the residuals are measured before the microgrid constitutive conversions, the implemented nonzero block obeys

$$\begin{aligned}\mathcal{K}_{B,C,L} &= \begin{bmatrix} 0 & B L_E^{-1} \\ -B^\top C_N^{-1} & 0 \end{bmatrix},\\ \|\mathcal{K}_{B,C,L}\|_M &\le \|B\|_2 \max\{\|L_E^{-1}\|_2, \|C_N^{-1}\|_2\}.\end{aligned}$$

Uniformly compatible coordinate changes add only their induced-norm factors; denote the resulting size-independent product by C_{inc}. For a simple undirected graph of maximum degree $d_{\max}$, $BB^\top$ is its graph Laplacian and $\|B\|_2 \le \sqrt{2 d_{\max}}$; hence $\|K(\mathcal{S})\|_M \le C_{\mathrm{inc}} \sqrt{2 d_{\max}}$. In the matched coordinates $C_{\mathrm{inc}} = 1$, so (75) contains the microgrid term $\|B_{\mathrm{inc}}(\mathcal{S})\|_2 \delta_{\mathrm{edge},\mathcal{S}}$ to the first power. A Laplacian power $B \operatorname{diag}(\cdot) B^\top$ would arise only after eliminating the explicit edge state and is not the operator considered here.

Theorem C.2 (Observer-aware deployment transfer)**.** *Assume* (A1) *incremental contraction,* $\|\widehat{F}_k(z) - \widehat{F}_k(z')\|_M \le q_k \|z - z'\|_M$ *on* $\Omega_{\mathcal{S}}$ *with* $q_k \le q \in (0,1]$*;* (A2) *state-map consistency,* $\|\Psi_{\Delta t}(z,u) - \varphi_{\Delta t}(z,u)\|_M \le K_\Psi \Delta t^3$*;* (A3) *commutation defect:* $\delta_k = \sup_{x \in \Omega_{\mathcal{S}}^X} \|\Pi_{\mathcal{S}} F_k(x) - \widehat{F}_k(\Pi_{\mathcal{S}} x)\|_M$ *obeys the bound of Lemma C.1 —* $\delta_k \le \Delta t\, e^{L_f \Delta t}\big(\delta_{\mathrm{loc}} + \|K(\mathcal{S})\|\, \delta_{\mathrm{edge}} + \delta_{\mathrm{port}}\big) + K_\Psi \Delta t^3$ — derived *(not assumed) from a pointwise transported field mismatch (its hypotheses H2–H4) together with (A2); for the edge-explicit microgrid* $\|K(\mathcal{S})\| = \|B_{\mathrm{inc}}(\mathcal{S})\|_2$ *to the first power;* (A4) *observer initialization* $e_0 \le \epsilon_{\mathrm{lift}}$ *(see Remark C.3);* (A5) *a matched mode/contact schedule with both trajectories in* $\Omega_{\mathcal{S}}$*. Then for every horizon* h*,*

$$e_h \ \le\ \Big(\textstyle\prod_{j=0}^{h-1} q_j\Big) e_0 + \sum_{k=0}^{h-1} \Big(\prod_{j=k+1}^{h-1} q_j\Big) \delta_k \ \le\ q^h\, \epsilon_{\mathrm{lift}} + S_h(q)\, \bar{\delta}, \tag{81}$$

with $\bar{\delta} = \max_k \delta_k$ *and* $S_h(q) = \frac{1-q^h}{1-q}$ *for* $q < 1$ *(and* h *for* $q = 1$*). If* $q < 1$ *the bound is uniform in* h*:* $\limsup_h e_h \le \bar{\delta}/(1-q)$.

Proof. By (A5), $\Pi_{\mathcal{S}} x_k \in \Omega_{\mathcal{S}}$, so the triangle inequality and (A1) give $e_{k+1} = \|\Pi_{\mathcal{S}} F_k x_k - \widehat{F}_k \hat{z}_k\|_M \le \|\Pi_{\mathcal{S}} F_k x_k - \widehat{F}_k(\Pi_{\mathcal{S}} x_k)\|_M + \|\widehat{F}_k(\Pi_{\mathcal{S}} x_k) - \widehat{F}_k \hat{z}_k\|_M \le \delta_k + q_k e_k$. Unrolling the scalar recursion $e_{k+1} \le q_k e_k + \delta_k$ gives the product-and-sum form; $q_j \le q$, $\delta_k \le \bar{\delta}$, and $\sum_{k=0}^{h-1} q^{h-1-k} = S_h(q)$ give the closed form. □

Remark C.3 (Source of the lift error)**.** The initialization ϵ_{lift} (A4) is a hypothesis of the deterministic bound, supplied by the deployed observer; we keep Theorem C.2 deterministic and source ϵ_{lift} at the observer layer rather than folding the observer's stochastics into the transfer bound. For a bounded-error (set-membership) observer, or under a declared bounded-noise assumption, ϵ_{lift} is a genuine deterministic bound. For a linear-Gaussian observer it is not: Gaussian errors have unbounded support, so a covariance gives no almost-sure finite bound. The honest replacement is a high-probability radius—with the stationary Kalman covariance P^* (the solution of the observer's algebraic Riccati equation) and zero bias, and $d = \dim e_0$,

$$\|e_0\|_M \le r(\alpha) := \sqrt{\chi^2_{d,\,1-\alpha}\,\lambda_{\max}\big(M^{1/2} P^* M^{1/2}\big)} \quad \text{with probability } \ge 1-\alpha.$$

Note that $r(\alpha)$ scales as a standard deviation $\sqrt{\lambda_{\max}(P^*)}$, not as the covariance P^* itself: in the scalar random-walk model with identity observation, $P^* = \sqrt{Q_\theta R} + \mathcal{O}(Q_\theta)$ (process noise Q_θ, observation noise R), so the radius is $(Q_\theta R)^{1/4}$, whereas the general linear case solves a matrix Riccati equation with no universal square-root form. Under repeated q-only refresh the single-step bound does not simply carry over: the faithful object is a stochastic recursion $e_{k+1} \le q_k e_k + \delta_k + \nu_k$ with ν_k the stepwise reset error, and its bounded-in-probability analysis is Theorem C.4 below. Theorem C.2 is thus the deterministic transfer bound with an assumed initialization, and Theorem C.4 its probabilistic instantiation for a refreshing Gaussian observer.

Theorem C.4 (Probabilistic deployment transfer under refreshed observation)**.** *Assume the hypotheses of Theorem C.2 with $q_k \le q \in (0,1)$ and $\delta_k \le \bar{\delta}$, and replace the fixed initialization (A4) by a* refreshed-*observation recursion*

$$e_{k+1} \le q_k e_k + \delta_k + \nu_k, \qquad e_0 = \nu_{-1}, \tag{82}$$

in which each step injects a stochastic reset error $\nu_k = \|\xi_k\|_M$ (the observer correction, gain included, mapped into the state metric) obeying the individual tail bound

$$\Pr\big[\nu_k > r(\alpha')\big] \le \alpha', \qquad r(\alpha') = \sqrt{\chi^2_{d,\,1-\alpha'}\,\lambda_{\max}\big(M^{1/2}\Sigma M^{1/2}\big)},$$

for $k = -1, 0, \dots, h-1$, where $\xi_k \sim \mathcal{N}(0,\Sigma)$ with zero bias and $d = \dim e_0$ (e.g. $\Sigma = P^*$ at stationarity). Then for every $\alpha \in (0,1)$ and horizon h, with $\alpha' = \alpha/(h+1)$,

$$\Pr\left[\max_{0\le j\le h} e_j \;\le\; S_h(q)\,\bar{\delta} + r\big(\tfrac{\alpha}{h+1}\big)\,S_{h+1}(q)\right] \;\ge\; 1-\alpha. \tag{83}$$

Under a summable schedule $\sum_{k\ge 0}\alpha_k = \alpha$ the per-step events combine over the whole trajectory: with probability $\ge 1-\alpha$, $e_k \le S_k(q)\,\bar{\delta} + r(\alpha_k)\,S_{k+1}(q)$ for every *k simultaneously. Because Gaussian refresh has $\sup_k \nu_k = \infty$ almost surely, this is a* growing *time-indexed envelope—$r(\alpha_k)$ inflating as $\sqrt{\log(1/\alpha_k)}$—not a bounded supremum.*

Proof. By the union (Boole) bound over the $h+1$ events $\{\nu_k > r(\alpha')\}$, $k = -1, \dots, h-1$,

$$\Pr\big[\exists\, k:\ \nu_k > r(\alpha')\big] \le (h+1)\alpha' = \alpha.$$

On the complementary event every $\nu_k \le r(\alpha')$. Unrolling (82) with $q_j \le q$, $\delta_j \le \bar{\delta}$, and $e_0 \le r(\alpha')$ gives, for each $j \le h$,

$$e_j \;\le\; q^j e_0 + \sum_{i=0}^{j-1} q^{j-1-i}\big(\delta_i + \nu_i\big) \;\le\; S_j(q)\,\bar{\delta} + r(\alpha')\Big(q^j + \textstyle\sum_{i=0}^{j-1} q^{j-1-i}\Big).$$

Since $q^j + S_j(q) = q^j + \frac{1-q^j}{1-q} = \frac{1-q^{j+1}}{1-q} = S_{j+1}(q)$ and S_j, S_{j+1} are nondecreasing in j, each $e_j \le S_h(q)\bar{\delta} + r(\alpha')S_{h+1}(q)$; the maximum over $j \le h$ obeys the same bound, which is (83). For the infinite-horizon claim, a schedule with $\sum_k \alpha_k = \alpha$ makes the union bound over all steps converge, giving the simultaneous per-step envelope $e_k \le S_k(q)\bar{\delta} + r(\alpha_k)S_{k+1}(q)$; the envelope grows through $r(\alpha_k) = \mathcal{O}(\sqrt{\log(1/\alpha_k)})$ and does not collapse to a bounded supremum, since $\sup_k \nu_k = \infty$ almost surely for Gaussian refresh. □

Remark C.5 (Tighter constant under independent refreshes)**.** If the ξ_k are independent, $\nu_k = \|\xi_k\|_M$ is sub-Gaussian about its mean $\mu = \mathbb{E}\|\xi_0\|_M$ with variance proxy $\sigma_M^2 = \lambda_{\max}(M^{1/2}\Sigma M^{1/2})$, because $\|\cdot\|_M$ is 1-Lipschitz in the M-metric and $M^{1/2}\xi_k$ is Gaussian (Gaussian concentration). The geometric sum $W_h = \sum_{i=0}^{h-1} q^{h-1-i}\nu_i$ then concentrates,

$$\Pr\big[W_h > \mu\, S_h(q) + \sigma_M \sqrt{2\, T_h\, \log(1/\alpha)}\big] \le \alpha, \qquad T_h = \tfrac{1-q^{2h}}{1-q^2},$$

replacing the union bound's $\sqrt{\log(h{+}1)}$ growth by a single $\sqrt{\log(1/\alpha)}$; the deterministic $S_h(q)\bar{\delta}$ and $q^h e_0$ terms are unchanged. This is the tighter constant when the per-step observer resets are (conditionally) independent.

Empirical grounding of the probabilistic transfer bound. We evaluate Theorem C.4 on the real affine deployment maps, using the certified metrics and contraction factors of Remark C.6. A stationary-gain, q-only Kalman observer used $Q_\theta = 10^{-6} I$ and $R = I$; $N = 3000$ independent rollouts of horizon $h = 200$ were drawn for each deployment. The table reports coverage of $\max_{0\le j\le h} e_j$ and the ratio of the bound to its empirical $(1-\alpha)$-quantile. The C.5-max columns use the simultaneous-max extension of Remark C.5, adding the standard maximum-of-sub-Gaussians expectation term. Every measured coverage was one, above the required Monte-Carlo lower band, but this is an empirical check and not a certificate. The C.4 envelope is conservative: its ratio grows from 17–18× on the oscillator to 31–33× on the near-unit-q microgrid. Independent-refresh concentration is tighter, reducing these ranges to 6–7× and 15–16×, respectively, while retaining the same measured coverage; neither result supports a claim of tightness.

Deployment	q	α	C.4 cov.	C.4 ratio	C.5-max cov.	C.5-max ratio
damped oscillator, $d_{\text{base}} = 3$	0.977848100	0.10	1.000	18.33×	1.000	7.10×
		0.05	1.000	17.82×	1.000	6.75×
		0.01	1.000	16.97×	1.000	6.12×
complete microgrid, $N = 6$	0.999632134	0.10	1.000	33.19×	1.000	16.43×
		0.05	1.000	32.46×	1.000	15.88×
		0.01	1.000	31.18×	1.000	14.87×

The contraction factor is verified, not structural. (A1) is load-bearing and does *not* follow from passivity or symplecticity: a symplectic step preserves a two-form, not a positive-definite norm, so its induced norm can exceed 1 (and with nontrivial Jordan structure it need not be nonexpansive in any fixed metric). The narrower true statement is that for the *specific* conservative linear oscillator the one-step operator has $\rho(A) = 1$, so no $q < 1$ is feasible in a fixed metric (Remark C.6); this does not generalize from symplecticity or passivity. We therefore establish q from the deployed one-step map in a metric on $\Omega_{\mathcal{S}}$, on untrained (KNOWN-regime) systems: for the *linear* deployments it is *interval-certified* by a semidefinite metric search with verified-Cholesky enclosure (Remark C.6), and for the nonlinear pendulum it is measured as a one-step operator norm. On a damped oscillator the linear contraction is $q < 1$ and metric-dependent. On the nonlinear pendulum the map is non-normal—the Euclidean one-step norm exceeds 1 at every state—yet the spectral radius is below 1 in the damped basin, and a Lyapunov metric (constructed from the discrete Lyapunov equation, *not* the energy Hessian, which the measurement shows is insufficient off the equilibrium) realizes $\|\widehat{F}\|_M < 1$ (measured 0.987–0.999). On coupled multi-block microgrids the contraction holds over the tested counts ($\rho < 1$ for $N \in \{3, 4, 6\}$, with margin roughly constant across these bounded-degree interconnections); this is evidence over the tested deployments, not a proof of asymptotic graph-size uniformity. The size-uniform bound of Lemma C.1 is moreover stated in the deployment-normalized RMS metric $M_{\mathcal{S}}$: a uniform per-block error is $\mathcal{O}(1)$ in $M_{\mathcal{S}}$ but $\mathcal{O}(\sqrt{b_{\mathcal{S}}})$ in the stacked Euclidean norm. The incidence power in (A3) is $\|B_{\text{inc}}\|^1$: because C-PHAST keeps line blocks explicit, the oriented incidence appears once per node–edge coupling block, not as a Laplacian $B_{\text{inc}}\,\text{diag}\,B_{\text{inc}}^\top$ (which would arise only after eliminating the edge dynamics). The state-consistency constant K_Ψ in (A2) is regime-specific in the same way: it is the explicit-leapfrog local-truncation constant for the separable-after-canonicalization deployments the benchmarks use (diagonal or block-diagonal mass), so a strongly configuration-dependent, nonseparable mass $M(q)$ falls outside this particular constant. Reproduction scripts accompany the paper.

Remark C.6 (Interval-certified contraction on the known-regime linear deployments)**.** For the untrained KNOWN-regime linear systems, assumption (A1) can be certified globally rather than estimated from sampled rollouts. Each deployed one-step map is affine by analytic construction, so its linear part $A = D\widehat{F}$ is constant on all of state space and hence on every operating region $\Omega_{\mathcal{S}}$. We obtain A by differentiating the deployed one-step map in float64; agreement of the

automatic-differentiation Jacobian at two states is retained only as an implementation regression check. For each deployment, a trace-minimizing semidefinite program produces a float64 metric witness on the ray

$$I \preceq M \preceq 10^3 I, \qquad A^\top M A \preceq q^2 M.$$

The program makes the lower normalization active. In nine rows, the exact float64 solver output lay at most 3.39×10^{-10} along the infeasible side of that boundary; there we multiplied M by a positive scalar no larger than 1.000000000339. This selects a strictly interior representative of the same metric ray and changes neither $\mathrm{cond}(M)$ nor the contraction inequality. Both normalization LMIs are then re-certified for the rescaled float64 matrix.

All subsequent operations embed the binary64 entries of A and M as exact dyadic numbers and use outward-rounded interval arithmetic at 80 decimal digits. For each of

$$S_L = M - I, \qquad S_U = 10^3 I - M, \qquad S_C = q_{\mathrm{cert}}^2 M - A^\top M A,$$

a verified interval Cholesky factorization succeeds not only for $S_\bullet$ but for $S_\bullet - \underline{\lambda}_\bullet I$. Thus the three displayed $\underline{\lambda}_\bullet$ are rigorous lower bounds on the corresponding minimum eigenvalues. The certified condition-number upper bound is

$$\mathrm{cond}(M) \le \frac{10^3 - \underline{\lambda}_U}{1 + \underline{\lambda}_L}.$$

Each q_{cert} is the smallest value on a 10^{-12} reporting grid at or above both the previously reported value and the factor recomputed from the current real map; all rows passed at that initial value, so no additional grid inflation was required.

Known-regime linear deployment	q_{cert}	$\mathrm{cond}(M) \le$	$\underline{\lambda}_L$	$\underline{\lambda}_U$	$\underline{\lambda}_C$
damped oscillator, $k=2$, $d_{\mathrm{base}}=0.1$	0.998751560550	1.969495	2.32829×10^{-10}	9.98030×10^{2}	1.59627×10^{-3}
damped oscillator, $k=2$, $d_{\mathrm{base}}=0.3$	0.996264009963	1.956402	6.71874×10^{-14}	9.98043×10^{2}	4.75340×10^{-3}
damped oscillator, $k=2$, $d_{\mathrm{base}}=1.0$	0.987654320988	2.292786	1.84430×10^{-11}	9.97707×10^{2}	1.60504×10^{-2}
damped oscillator, $k=2$, $d_{\mathrm{base}}=3.0$	0.977848100000	5.408350	2.32830×10^{-10}	9.94591×10^{2}	4.40487×10^{-2}
microgrid, chain, $N=3$	0.999380844000	3.840699	1.92737×10^{-12}	9.96159×10^{2}	6.54601×10^{-4}
microgrid, chain, $N=4$	0.999413744000	3.863131	2.32831×10^{-10}	9.96136×10^{2}	6.20537×10^{-4}
microgrid, chain, $N=6$	0.999458282000	3.659778	2.32830×10^{-10}	9.96340×10^{2}	5.70189×10^{-4}
microgrid, ring, $N=3$	0.999378702000	3.925462	2.32830×10^{-10}	9.96074×10^{2}	6.59850×10^{-4}
microgrid, ring, $N=4$	0.999494825000	3.563284	2.32830×10^{-10}	9.96436×10^{2}	5.31468×10^{-4}
microgrid, ring, $N=6$	0.999496890000	3.483352	2.32831×10^{-10}	9.96516×10^{2}	5.28510×10^{-4}
microgrid, complete, $N=3$	0.999378702000	3.921662	2.32830×10^{-10}	9.96078×10^{2}	6.59850×10^{-4}
microgrid, complete, $N=4$	0.999511950000	3.414291	5.33913×10^{-11}	9.96585×10^{2}	5.12790×10^{-4}
microgrid, complete, $N=6$	0.999632134000	3.611252	2.32830×10^{-10}	9.96388×10^{2}	3.81262×10^{-4}

The oscillator step is at $\Delta t = 0.05$ and the microgrid step at $\Delta t = 0.01$; the latter matrices are extracted at the discrete DC fixed points of the real affine maps. Since every $q_{\mathrm{cert}} < 1$ and $S_C \succ 0$, the affine identity $\widehat{F}(z) - \widehat{F}(z') = A(z - z')$ gives

$$\|\widehat{F}(z) - \widehat{F}(z')\|_M \le q_{\mathrm{cert}} \|z - z'\|_M$$

for all states and therefore discharges (A1) of Theorem C.2 on any $\Omega_{\mathcal{S}}$ for these deployments.

The conservative statement remains deliberately narrower. For the real conservative linear-oscillator step ($k = 1$, $\Delta t = 0.05$), $\rho(A) = 1$. Because $A^\top M A \preceq q^2 M$ with $M \succ 0$ implies $\rho(A) \le q$, no $q < 1$ is feasible; the limiting value is $q = 1$. The interval certificates above cover only the analytic damped linear deployments. They do not certify the nonlinear pendulum on a region, and trained models remain outside this certificate.

Remark C.7 (Certified state-consistency constants)**.** The consistency constant K_Ψ of (A2) is likewise interval-certified, not fitted, for the analytic KNOWN-regime instances. With $\Psi_{\Delta t}$ the deployed second-order step and $\varphi_{\Delta t}$ the exact flow on $\Omega = [-1, 1]^2$, $0 < \Delta t \le 0.16$, outward-rounded interval evaluation of the third-order Taylor remainder gives $\|\Psi_{\Delta t} - \varphi_{\Delta t}\| \le K_\Psi \Delta t^3$ with $K_\Psi = 0.197$ for the undamped harmonic oscillator and 0.240 for the nonlinear pendulum. For the damped oscillator, the second-order Cayley (A-stable) damping half-step keeps the Δt^2 coefficient of $\Psi_{\Delta t} - \varphi_{\Delta t}$

identically zero—a first-order forward-Euler half would instead leave an $\mathcal{O}(\Delta t^2)$ state defect—and the same interval method certifies a finite K_Ψ, re-derived for the Cayley step alongside the contraction factors of Remark C.6. These are the explicit-leapfrog constants of the separable regime; configuration-dependent mass and trained modules lie outside the certified set.

Metric reconciliation. These K_Ψ are certified in the Euclidean metric of their analytic instances ($M = I_2$), whereas (A2) uses the deployment metric $M_{\mathcal{S}}$; the conversion is $K_{\Psi,M_{\mathcal{S}}} \le \sqrt{\lambda_{\max}(M_{\mathcal{S}})}\, K_{\Psi,2}$. Likewise, Theorem C.2 bounds the state error in $M_{\mathcal{S}}$ while the readout constants of Lemma D.3 use a reliability metric M_{rel}; composing them carries the factor $c_{M_{\mathrm{rel}}\leftarrow M_{\mathcal{S}}} = \sqrt{\lambda_{\max}(M_{\mathcal{S}}^{-1/2} M_{\mathrm{rel}} M_{\mathcal{S}}^{-1/2})}$, so the per-channel bound of Proposition D.1 reads $|\hat{r} - r| \le L_m^x\, c_{M_{\mathrm{rel}}\leftarrow M_{\mathcal{S}}}\, \beta_k + \epsilon_m^{\mathrm{map}}$. A single metric throughout ($M_{\mathrm{rel}} = M_{\mathcal{S}}$) makes both factors unity.

D. Typed-Consequence Decision Reliability

Given the state tube of Theorem C.2, each typed consequence—and the planner decisions built on it—inherits a computable margin whenever the state-tube, readout, map-error, and aggregation constants are valid upper bounds on the operating region. Where those constants are certified (the analytic KNOWN-regime channels) the margin is a certificate; where they are empirically calibrated (trained and hybrid deployments) it is a calibrated bound. This turns the typed-consequence interface into a decision object with *conditional* guarantees.

Setup. For candidate a, channel m, step k: truth $r_{m,k}^a = o_{m,\mathcal{S}}(x_k^a, u_k^a)$ and prediction $\hat{r}_{m,k}^a = \hat{o}_{m,\mathcal{S}}(\hat{z}_k^a, u_k^a)$; Theorem C.2 gives $\|\Pi_{\mathcal{S}} x_k^a - \hat{z}_k^a\|_M \le \beta_k^a$.

Proposition D.1 (Channel and horizon reliability)**.** *Assume* (B1) *per-channel* $|\hat{r}_{m,k}^a - r_{m,k}^a| \le L_m^x \beta_k^a + \epsilon_m^{\mathrm{map}}$ *for the paired states* $x_k^a \in \Pi_{\mathcal{S}}^{-1}(\Omega_{\mathcal{S}})$ *and* $\hat{z}_k^a \in \Omega_{\mathcal{S}}$ *(with* $\|\Pi_{\mathcal{S}} x_k^a - \hat{z}_k^a\|_M \le \beta_k^a$*), where* L_m^x *is a certified readout-Lipschitz constant (Lemma D.3) and* $\epsilon_m^{\mathrm{map}}$ *the learned-vs-physical readout error;* (B2) *channels smooth or mode-conditioned (energy, dissipation, port work, voltage/current directly; contact/switching identities only within a fixed mode or via a smooth surrogate);* (B3) *a* $\kappa_{m,H}$*-Lipschitz* (ℓ_p) *horizon aggregator* $\rho_m^a = A_{m,H}(r_{m,0:H}^a)$*. Then*

$$\left|\hat{\rho}_m^a - \rho_m^a\right| \;\le\; \kappa_{m,H} \left\| \left(L_m^x \beta_k^a + \epsilon_m^{\mathrm{map}}\right)_{k=0}^{H} \right\|_p \;=:\; \Delta_m(a). \tag{84}$$

Proof. Apply (B1) pointwise in k, then the $\kappa_{m,H}$-Lipschitz aggregator (B3) to the horizon vector; β_k^a is the state tube of Theorem C.2. □

Theorem D.2 (Decision margins)**.** *Let* $\mathcal{A}$ *be a nonempty candidate set on which the displayed extrema and minimizers exist. Suppose Proposition D.1 gives, for every* $a \in \mathcal{A}$ *and channel* $m = 1, \dots, M$*,*

$$|\hat{\rho}_m(a) - \rho_m(a)| \le \Delta_m(a). \tag{85}$$

Write

$$\Delta_\rho(a) := (\Delta_m(a))_{m=1}^M \in \mathbb{R}_+^M.$$

Set

$$J_w(a) = w^\top \rho(a), \qquad \hat{J}_w(a) = w^\top \hat{\rho}(a), \qquad \Delta_J(a) = \sum_{m=1}^M |w_m|\, \Delta_m(a).$$

Assume in addition (B4) *for every hard-constraint index* j *there is a candidate-level map* $\mathrm{g}_j(a, \cdot)$ *such that*

$$g_j(a) = \mathrm{g}_j(a, \rho(a)), \qquad \hat{g}_j(a) = \mathrm{g}_j(a, \hat{\rho}(a)),$$

and, uniformly over $a \in \mathcal{A}$*, this map is* L_{g_j}*-Lipschitz in its consequence argument in* ℓ_p *on a set containing the relevant true and predicted consequence vectors:*

$$|\mathrm{g}_j(a, r) - \mathrm{g}_j(a, r')| \le L_{g_j} \|r - r'\|_p. \tag{86}$$

Define

$$\Delta_{g_j}(a) := L_{g_j} \|\Delta_\rho(a)\|_p. \tag{87}$$

Then $|\hat{J}_w(a) - J_w(a)| \le \Delta_J(a)$*, and* (i) *the ranking of* a, b *is preserved if* $|\hat{J}_w(a) - \hat{J}_w(b)| > \Delta_J(a) + \Delta_J(b)$*;* (ii) *the predicted minimizer has true regret at most* $2 \max_a \Delta_J(a)$*;* (iii) *on a finite candidate set the predicted and true argmin coincide if the true best-vs-second gap exceeds* $2 \max_a \Delta_J(a)$*;* (iv) *a constraint* $g_j(a) \le 0$ *is certified by* $\hat{g}_j(a) \le -\Delta_{g_j}(a)$*.*

Proof. The coordinatewise envelope (85) gives

$$\|\hat{\rho}(a) - \rho(a)\|_p \le \|\Delta_\rho(a)\|_p. \tag{88}$$

Consequently,

$$|\hat{J}_w(a) - J_w(a)| \le \sum_{m=1}^{M} |w_m|\,|\hat{\rho}_m(a) - \rho_m(a)| \le \Delta_J(a).$$

For (i), set $\hat{d} = \hat{J}_w(a) - \hat{J}_w(b)$ and $d = J_w(a) - J_w(b)$. The score envelope implies

$$|d - \hat{d}| \le \Delta_J(a) + \Delta_J(b).$$

Thus $|\hat{d}| > \Delta_J(a) + \Delta_J(b)$ forces d and $\hat{d}$ to have the same sign.

For (ii), let $\hat{a} \in \arg\min_{a\in\mathcal{A}} \hat{J}_w(a)$ and $a^\star \in \arg\min_{a\in\mathcal{A}} J_w(a)$. Predicted optimality and the two endpoint envelopes give

$$\begin{aligned} J_w(\hat{a}) - J_w(a^\star) &\le \hat{J}_w(\hat{a}) + \Delta_J(\hat{a}) - \hat{J}_w(a^\star) + \Delta_J(a^\star) \\ &\le \Delta_J(\hat{a}) + \Delta_J(a^\star) \le 2\max_{a\in\mathcal{A}} \Delta_J(a). \end{aligned}$$

For (iii), let $a^\star$ be the unique true minimizer and let the true best-vs-second gap exceed $2D$, where $D = \max_{a\in\mathcal{A}} \Delta_J(a)$. For every $a \ne a^\star$,

$$\hat{J}_w(a) - \hat{J}_w(a^\star) \ge J_w(a) - J_w(a^\star) - \Delta_J(a) - \Delta_J(a^\star) > 0,$$

so $a^\star$ is also the unique predicted minimizer.

Finally, (B4) and (88) give the missing constraint-error bound

$$|\hat{g}_j(a) - g_j(a)| \le L_{g_j}\|\hat{\rho}(a) - \rho(a)\|_p \le \Delta_{g_j}(a). \tag{89}$$

Hence $\hat{g}_j(a) \le -\Delta_{g_j}(a)$ implies $g_j(a) \le \hat{g}_j(a) + \Delta_{g_j}(a) \le 0$, proving (iv). □

Empirical grounding. For each smooth consequence channel o_m, the quantity used in Proposition D.1 is a uniform certified upper bound

$$L_m^x \;\ge\; \sup_{(z,u)\in\Omega\times U} \|\nabla_z o_m(z,u)\|_{M^{-1}},$$

on a stated compact convex operating region Ω and, for port work, a stated compact input set U; it is not a test-set average. Thus $|o_m(z,u) - o_m(z',u)| \le L_m^x\|z - z'\|_M$ for all $z, z' \in \Omega$ at the same $u \in U$. Here $M \succ 0$ is the fixed metric used by the reliability bound, not the Hamiltonian's physical mass. For a quadratic Hamiltonian $H(z) = \frac{1}{2}z^\top Q z$, with $Q = \nabla^2 H$, the correct statements are

$$\begin{aligned} L_{H,\star}^x &:= \sup_{z\in\Omega} \|Qz\|_{M^{-1}}, \\ L_{H,\star}^x &\le \|M^{-1/2} Q M^{-1/2}\|_2 \sup_{z\in\Omega} \|z\|_M. \end{aligned}$$

L_H^x used in the proposition must satisfy $L_H^x \ge L_{H,\star}^x$. $\|Q\|_2 := \sup_{v\ne 0} \|Qv\|_2/\|v\|_2$ is the spectral norm, and a Hessian norm by itself is not L_H^x. If $e = \nabla H$ and $d = e^\top R(z)e$, then $\nabla d = 2(\nabla^2 H)^\top Re + r_R$ with $(r_R)_i = e^\top(\partial_i R)e$, so certification must include $\nabla^2 H$, R, and ∇R. Likewise, for $P = e^\top B(z)u$, one has $\nabla_z P = (\nabla^2 H)^\top B(z)u + b_B$ with $(b_B)_i = e^\top(\partial_i B)u$, and the bound must include B, ∇B, and the full set U. Linear electrical readouts $c^\top z + b(u)$ have the uniform constant $\|c\|_{M^{-1}}$, but this does not make a generally nonlinear known-physics model affine. Lemma D.3 records these formulas, an interval-certified smooth DGU instance, and the exact horizon-aggregator constants. The separate quantity ε^{map} must likewise be supplied as a uniform supremum bound over its declared region; semi-synthetic perturbations can diagnose it but do not certify it, and no ε^{map} certificate is made here. Contact, switching, and discrete support-count channels are excluded from this smooth statement and require mode-conditional constants.

Lemma D.3 (Certified Lipschitz constants for smooth readouts and horizon aggregators)**.** *Let $\Omega \subset \mathbb{R}^n$ be a nonempty compact convex box, let $U \subset \mathbb{R}^r$ be a compact input box, and let $M_{\text{rel}} \succ 0$ be a fixed reliability metric. Write*

$$\|v\|_{M_{\text{rel}}} := (v^\top M_{\text{rel}} v)^{1/2}, \qquad \|a\|_{M_{\text{rel}}^{-1}} := (a^\top M_{\text{rel}}^{-1} a)^{1/2}.$$

This M_{rel} is not the configuration-dependent physical mass $\mathcal{M}(q)$ appearing in a Hamiltonian. If $o(z,u)$ is continuously differentiable on a neighbourhood of $\Omega \times U$, then any outward-rounded interval enclosure

$$L^x_{o,\mathrm{cert}} \;\geq\; \sup_{(z,u)\in\Omega\times U} \|\nabla_z o(z,u)\|_{M_{\mathrm{rel}}^{-1}} \tag{90}$$

is a uniform constant satisfying, for every $z, z' \in \Omega$ and every $u \in U$,

$$|o(z,u) - o(z',u)| \leq L^x_{o,\mathrm{cert}} \|z - z'\|_{M_{\mathrm{rel}}}.$$

In particular, the following are valid smooth-channel certificates.

1. *If $H(z) = \frac{1}{2} z^\top Q z$ with $Q = Q^\top = \nabla^2 H$, then*

$$L^x_{H,\mathrm{cert}} \geq L^x_{H,\star} := \sup_{z\in\Omega} \|Qz\|_{M_{\mathrm{rel}}^{-1}}, \qquad L^x_{H,\star} \leq \|M_{\mathrm{rel}}^{-1/2} Q M_{\mathrm{rel}}^{-1/2}\|_2 \sup_{z\in\Omega} \|z\|_{M_{\mathrm{rel}}}.$$

   Here and below $\|A\|_2 := \sup_{v\neq 0} \|Av\|_2/\|v\|_2$ is the spectral norm (largest singular value); in particular, an unqualified $\|Q\|$ means $\|Q\|_2$. The Hessian norm alone is not a Lipschitz constant for the scalar readout H on an unbounded domain.

2. *Let $e(z) := \nabla H(z)$, $Q(z) := \nabla^2 H(z)$, and let $R(z) = R(z)^\top \succeq 0$. For $d(z) := e(z)^\top R(z) e(z)$,*

$$\nabla d(z) = 2Q(z)^\top R(z) e(z) + r_R(z), \qquad [r_R(z)]_i = e(z)^\top (\partial_i R(z)) e(z),$$

   and hence

$$L^x_{d,\mathrm{cert}} \geq \sup_{z\in\Omega} \|2Q(z)^\top R(z) e(z) + r_R(z)\|_{M_{\mathrm{rel}}^{-1}}.$$

   Thus a dissipation certificate generally depends on $\nabla^2 H$, R, and ∇R.

3. *For the port-work readout $P(z,u) := e(z)^\top B(z) u$, define $[b_B(z,u)]_i := e(z)^\top (\partial_i B(z)) u$. Then*

$$\nabla_z P(z,u) = Q(z)^\top B(z) u + b_B(z,u),$$

   and

$$L^x_{P,\mathrm{cert}} \geq \sup_{(z,u)\in\Omega\times U} \|Q(z)^\top B(z) u + b_B(z,u)\|_{M_{\mathrm{rel}}^{-1}}.$$

   State dependence of B and the declared input set U therefore cannot be omitted. For a linear electrical channel $o(z,u) = c^\top z + b(u)$, the global valid constant is $L^x_o = \|c\|_{M_{\mathrm{rel}}^{-1}}$.

For the registered PHAST KNOWN*-regime DGU, the model gives*

$$H(q,p) = \tfrac{1}{2}\left(kq^2 + \frac{p^2}{\mathcal{M}}\right), \quad e = (kq, p/\mathcal{M})^\top, \quad R = \mathrm{diag}(0,r), \quad B = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$

where the columns of B are ordered as $(u_{\mathrm{buck}}, u_{\mathrm{load}})$ and the model parameters are $k = 0.4545$, $\mathcal{M} = 1.8$, and $r = 1.2$. The observable-state map has $s_q = 2.2$ and $s_p = 1.8$, so its direct electrical channels are $V = q/s_q$ and $I = p/s_p$; these are kept distinct from the Hamiltonian efforts kq and $p/\mathcal{M}$. Take $M_{\mathrm{rel}} = I_2$ and the review boxes

$$\Omega = \{(q,p) : 18 \leq q/s_q \leq 30,\ 0 \leq p/s_p \leq 5\},$$
$$U = [22.5, 25.5] \times [-3.12, -1.68].$$

Outward-rounded interval evaluation of the symbolic gradients on this full box gives the following non-vacuous certified choices:

$$L^x_{H,\mathrm{cert}} = 30.413812, \quad L^x_{d,\mathrm{cert}} = 6.666668, \quad L^x_{P,\mathrm{cert}} = 14.237476,$$
$$L^x_{V,\mathrm{cert}} = 0.454546, \quad L^x_{I,\mathrm{cert}} = 0.555556.$$

for $d = r(p/\mathcal{M})^2$, $P = (p/\mathcal{M})u_{\mathrm{buck}} + (kq)u_{\mathrm{load}}$, $V = q/s_q$, and $I = p/s_p$. The displayed decimals are rounded upward from the interval endpoints. These are conditional certificates on the stated boxes, not a claim that Ω is forward invariant.

For completeness, let a horizon contain $H \geq 1$ transitions, let $r_{1:H}$ denote its H stage readouts, let $h_{0:H}$ denote its $H+1$ endpoint readouts, and measure perturbations in ℓ_p, $1 \leq p \leq \infty$. Let p^* be the dual exponent, $1/p + 1/p^* = 1$ (with the usual endpoint conventions). The exact aggregator constants are as follows. For the ledger row, use the explicit signature

$$\mathcal{R}_H(h_0, h_H, d_{1:H}, w_{1:H}) := h_H - h_0 + \sum_{k=1}^{H} d_k - \sum_{k=1}^{H} w_k.$$

aggregator	*explicit signature*	$\kappa_{m,H}$ *in* ℓ_p
terminal	$h_{0:H} \mapsto h_H$	1
weighted linear	$r_{1:H} \mapsto \sum_{k=1}^{H} w_k r_k$	$\|w\|_{p^*}$
sum	$r_{1:H} \mapsto \sum_{k=1}^{H} r_k$	H^{1/p^*}
rectangle integral	$r_{1:H} \mapsto \Delta t \sum_{k=1}^{H} r_k$	$\Delta t\, H^{1/p^*}$
mean	$r_{1:H} \mapsto H^{-1} \sum_{k=1}^{H} r_k$	$H^{-1/p}$
maximum or minimum	$r_{1:H} \mapsto \max_k r_k$ *or* $\min_k r_k$	1
energy change	$h_{0:H} \mapsto h_H - h_0$	$2^{1/p^*}$
ledger residual	$\mathcal{R}_H$	$(2+2H)^{1/p^*}$

If an equal-weight sum, rectangle integral, or mean consumes all $H+1$ endpoint values $r_{0:H}$ rather than H stage values, every occurrence of H in that row is replaced by $H+1$ (the maximum/minimum constant remains 1). The same $2^{1/p^*}$ bound applies after an absolute-value or positive-part map of the energy change. For trapezoidal quadrature on H intervals,

$$\kappa_{m,H} = \Delta t\big(H - 1 + 2^{1-p^*}\big)^{1/p^*} \quad (p^* < \infty),$$

while for $p^* = \infty$ it is Δt when $H \geq 2$ and $\Delta t/2$ when $H = 1$. In particular, the sum has $\kappa = 1$ in ℓ_1, $\sqrt{H}$ in ℓ_2, and H in ℓ_∞; if every stage error is bounded by the same scalar, its accumulated sum still grows linearly in H. If the ledger accepts already-aggregated scalars (h_0, h_H, D, W), its constant is instead $4^{1/p^*}$; a software reduction over additional keys or tensor entries must state that enlarged signature and coefficient vector.

All claims above concern differentiable readouts within one smooth mode. Contact, switching, and discrete support-count channels are excluded and require mode-conditional treatment. No bound on $\varepsilon^{\mathrm{map}}$ is asserted here.

Proof. For fixed u, convexity of Ω and the line-segment fundamental theorem of calculus give

$$o(z,u) - o(z',u) = \int_0^1 \nabla_z o(z' + t(z-z'), u)^\top (z - z')\, dt.$$

Cauchy–Schwarz in the dual pair $(\|\cdot\|_{M_{\mathrm{rel}}^{-1}}, \|\cdot\|_{M_{\mathrm{rel}}})$ proves (90). Direct differentiation gives the displayed energy, dissipation, and port-work gradients. The numerical bounds are outward interval enclosures of every operation in those symbolic gradients; therefore they bound the continuum boxes, not merely sampled points. Finally, Hölder's inequality gives $\|a\|_{p^*}$ for every linear aggregator with coefficient vector a, and the constants are sharp by duality. The maximum/minimum maps are 1-Lipschitz because $|\max r - \max s| \leq \|r - s\|_\infty \leq \|r - s\|_p$, with the same argument for the minimum. □

Corollary D.4 (Lipschitz full-objective margins)**.** *Let $\mathcal{J} : \mathcal{R} \subset \mathbb{R}^M \to \mathbb{R}$ be L_J-Lipschitz in ℓ_p on a set containing $\rho(a)$ and $\hat{\rho}(a)$ for every $a \in \mathcal{A}$, and define*

$$J(a) = \mathcal{J}(\rho(a)), \qquad \hat{J}(a) = \mathcal{J}(\hat{\rho}(a)), \qquad \Delta_J(a) := L_J \|\Delta_\rho(a)\|_p.$$

Under the channel envelope (85),

$$|\hat{J}(a) - J(a)| \leq \Delta_J(a). \tag{91}$$

Therefore the ranking and argmin conclusions in Theorem D.2(i)–(iii) hold verbatim for this full objective after using the displayed Δ_J.

In particular, the full objective in Eq. (1) is covered channel by channel as follows. For the terminal channel, set

$$\rho_G(a) := \Psi_G(x_H(a)), \qquad \hat{\rho}_G(a) := \Psi_G(\hat{x}_H(a)). \tag{92}$$

If the terminal readout has a uniform constant L_G^x satisfying Lemma D.3 on the declared operating region and a uniform readout-map error $\epsilon_G^{\mathrm{map}}$, then the terminal aggregator has $\kappa_{G,H} = 1$. Because this selector uses only the final coordinate, the direct (B1) bound gives the valid, sharper envelope

$$\Delta_G(a) := L_G^x \beta_H^a + \epsilon_G^{\mathrm{map}}. \tag{93}$$

The generic Proposition-D.1 ℓ_p envelope over a padded endpoint sequence is also valid but can be looser. For each normalized factor q, regard the composition

$$o_q(z,u;\mathcal{S}_\kappa) := \bar{\ell}_q(\mathcal{O}_{\mathrm{target}}(z,u;\mathcal{S}_\kappa))\,, \qquad \hat{o}_q(z,u;\mathcal{S}_\kappa) := \bar{\ell}_q\Big(\widehat{\mathcal{O}}_{\mathrm{target}}(z,u;\mathcal{S}_\kappa)\Big) \tag{94}$$

as an additional scalar consequence channel. Assume it is smooth on a declared, mode-conditioned region. Let L_q^x satisfy the uniform condition of Lemma D.3, and define the rectangle-integrated channel by

$$\rho_q^{\mathrm{obj}}(a) := \Delta t \sum_{\ell=0}^{H-1} o_q(x_{\ell+1}(a), u_\ell; \mathcal{S}_\kappa), \tag{95}$$

$$\hat{\rho}_q^{\mathrm{obj}}(a) := \Delta t \sum_{\ell=0}^{H-1} \hat{o}_q(\hat{z}_{\ell+1}(a), u_\ell; \mathcal{S}_\kappa). \tag{96}$$

With p^* the dual exponent, Lemma D.3 gives the rectangle-integrator constant $\kappa_{q,H} = \Delta t\, H^{1/p^*}$, and Proposition D.1 gives the per-factor envelope

$$\Delta_q(a) := \Delta t\, H^{1/p^*} \left\| \left(L_q^x \beta_{\ell+1}^a + \epsilon_q^{\mathrm{map}}\right)_{\ell=0}^{H-1} \right\|_p. \tag{97}$$

Thus, for the augmented consequence vector

$$\rho^{\mathrm{obj}}(a) := \big(\rho_G(a), (\rho_q^{\mathrm{obj}}(a))_q\big), \qquad \Delta_\rho^{\mathrm{obj}}(a) := \big(\Delta_G(a), (\Delta_q(a))_q\big),$$

the true objective and its predicted counterpart are exactly

$$J_w(a) = \rho_G(a) + \sum_q w_q \rho_q^{\mathrm{obj}}(a), \qquad \hat{J}_w(a) = \hat{\rho}_G(a) + \sum_q w_q \hat{\rho}_q^{\mathrm{obj}}(a). \tag{98}$$

The model-generated quantity denoted J_w in Eq. (1) is $\hat{J}_w$ in this reliability comparison; J_w above is its true-trajectory comparator. Thus Eq. (1) is covered explicitly, rather than being treated as a linear score in the original untransformed consequences. Its linear consumer has ℓ_p Lipschitz modulus $L_J = \|(1,w)\|_{p^*}$, so (91) applies with $\Delta_\rho^{\mathrm{obj}}$. The componentwise linear bound

$$|\hat{J}_w(a) - J_w(a)| \le \Delta_G(a) + \sum_q |w_q| \Delta_q(a) \tag{99}$$

is also available and can be sharper. Hard constraints are not absorbed into this scalar objective; they are handled by (B4) and Proposition D.5 below.

Proof. The Lipschitz hypothesis and (88) give

$$|\hat{J}(a) - J(a)| \le L_J \|\hat{\rho}(a) - \rho(a)\|_p \le L_J \|\Delta_\rho(a)\|_p.$$

Theorem D.2(i)–(iii) uses only such a pointwise objective-error envelope, so its arguments apply unchanged. The channel envelopes for Eq. (1) then follow term by term from Proposition D.1: the terminal channel with selector constant $\kappa_{G,H} = 1$ gives (93); each factor channel with rectangle-integrator constant $\kappa_{q,H} = \Delta t\, H^{1/p^*}$ gives (97); and the linear identity (98) with modulus $L_J = \|(1,w)\|_{p^*}$ gives (99). □

Proposition D.5 (Robust feasibility and constrained regret)**.** *Let $\mathcal{A}$ be a nonempty finite candidate set. Assume the pointwise objective envelope*

$$|\hat{J}(a) - J(a)| \le \Delta_J(a) \tag{100}$$

and the constraint envelopes (89)*. The objective envelope may come from Theorem D.2 or from Corollary D.4. Define*

$$\mathcal{A}_{\text{feas}} := \{a \in \mathcal{A} : g_j(a) \le 0 \text{ for every } j\}, \tag{101}$$

$$\widehat{\mathcal{A}}_{\text{safe}} := \{a \in \mathcal{A} : \hat{g}_j(a) + \Delta_{g_j}(a) \le 0 \text{ for every } j\}. \tag{102}$$

Then every $a \in \widehat{\mathcal{A}}_{\text{safe}}$ is truly feasible; equivalently,

$$\widehat{\mathcal{A}}_{\text{safe}} \subseteq \mathcal{A}_{\text{feas}}. \tag{103}$$

If $\widehat{\mathcal{A}}_{\text{safe}} \neq \varnothing$ and

$$\hat{a}_{\text{safe}} \in \arg\min_{a \in \widehat{\mathcal{A}}_{\text{safe}}} \hat{J}(a), \qquad a^\star_{\text{safe}} \in \arg\min_{a \in \widehat{\mathcal{A}}_{\text{safe}}} J(a),$$

then the constrained regret relative to the best candidate in the same robust set satisfies

$$J(\hat{a}_{\text{safe}}) - J(a^\star_{\text{safe}}) \le 2 \max_{a \in \widehat{\mathcal{A}}_{\text{safe}}} \Delta_J(a) \le 2 \max_{a \in \mathcal{A}} \Delta_J(a). \tag{104}$$

For comparison with the optimum over the complete true feasible set, assume there exists

$$a^\star_{\text{feas}} \in \arg\min_{a \in \mathcal{A}_{\text{feas}}} J(a) \tag{105}$$

satisfying the true optimum-retention margin

$$g_j(a^\star_{\text{feas}}) \le -2\Delta_{g_j}(a^\star_{\text{feas}}) \qquad \text{for every } j. \tag{106}$$

Then $a^\star_{\text{feas}} \in \widehat{\mathcal{A}}_{\text{safe}}$, and

$$J(\hat{a}_{\text{safe}}) - J(a^\star_{\text{feas}}) \le 2 \max_{a \in \widehat{\mathcal{A}}_{\text{safe}}} \Delta_J(a). \tag{107}$$

Without membership of at least one true optimum in $\widehat{\mathcal{A}}_{\text{safe}}$—for example, without (106)*—no $2\Delta_J$ regret bound relative to the optimum over all of $\mathcal{A}_{\text{feas}}$ follows: safety tightening may exclude every true optimum. If $\widehat{\mathcal{A}}_{\text{safe}}$ is empty, the proposition certifies no selectable candidate.*

Proof. If $a \in \widehat{\mathcal{A}}_{\text{safe}}$, then for every j,

$$g_j(a) \le \hat{g}_j(a) + \Delta_{g_j}(a) \le 0,$$

which proves (103). Predicted optimality within the fixed set $\widehat{\mathcal{A}}_{\text{safe}}$ and the two endpoint objective envelopes give

$$\begin{aligned} J(\hat{a}_{\text{safe}}) - J(a^\star_{\text{safe}}) &\le \hat{J}(\hat{a}_{\text{safe}}) + \Delta_J(\hat{a}_{\text{safe}}) - \hat{J}(a^\star_{\text{safe}}) + \Delta_J(a^\star_{\text{safe}}) \\ &\le \Delta_J(\hat{a}_{\text{safe}}) + \Delta_J(a^\star_{\text{safe}}), \end{aligned}$$

which proves (104).

Under (106), the symmetric constraint envelope yields

$$\hat{g}_j(a^\star_{\text{feas}}) + \Delta_{g_j}(a^\star_{\text{feas}}) \le g_j(a^\star_{\text{feas}}) + 2\Delta_{g_j}(a^\star_{\text{feas}}) \le 0.$$

Thus the true optimum belongs to the robust set. Together with $\widehat{\mathcal{A}}_{\text{safe}} \subseteq \mathcal{A}_{\text{feas}}$, this means the best true objective value on the two sets is the same, so (104) gives (107). □

Scope of the constants. The quantities L_{g_j} and L_J above are stated hypotheses (or algebraic consequences of a fixed linear consumer), and the Δ_m are conditional consequences of Proposition D.1 using the readout/aggregator hypotheses of Lemma D.3. No numerical value for L_{g_j}, L_J, or a new Δ_m is certified here. In particular, a normalized factor must have a denominator bounded away from zero and a uniform gradient enclosure on its declared region before its L^x_q can be used. Contact, switching, discrete support-count, and unmatched-mode channels remain mode-conditional as in Proposition D.1; this block does not turn sampled slopes or average errors into deterministic bounds.

E. Energy–Casimir Control Benchmarks under q-only Deployment

Appendix A proves the exact-port stability statement. This appendix tests the implementation boundary that matters for C-PHAST: the controller must often run from q-only measurements, so port velocity is either given as an oracle reference or estimated from recent position history. The experiments answer two questions. First, on a single joint, does the Energy–Casimir controller stabilize the task under the same finite-step rollout protocol used by the paper? Second, after those gains are chosen once, can the same per-joint controller be copied onto coupled arms with $N \in \{2, 4, 8\}$ joints without retuning? The q-only rows should therefore be read through the observer-error calculation in Remark A.3, not as additional exact-stability theorems.

E.1. Benchmark Protocol and Evaluation Setup

The control benchmark is deliberately separate from the forecasting benchmarks in Table 6. Its purpose is not to compare world-model prediction error, but to test whether the pH port interface exposed by C-PHAST can drive a standard storage-shaping controller under realistic observation constraints.

Stage 1: Single-joint validation. A damped pendulum is the base case where the theory in Appendix A can be read directly: one plant port, one controller storage term, and one desired equilibrium. We use inertia I_p=1, mass m=1, length ℓ=1, gravity g=9.81, natural damping d_{nat}=0.1, and target $q^\star$=0. The controller is evaluated from five initial-condition regimes, namely near-stable, near-unstable, far-stable, far-unstable, and high-energy, with 20 trials per regime.

Stage 2: Controller attachment across tested arm counts. The same per-joint gains from Stage 1 are then copied onto coupled planar arms with $N \in \{2, 4, 8\}$ joints. The coupling is skew-symmetric, so Appendix A predicts that coupling should exchange power between joints without changing the total shaped-storage dissipation calculation. No controller parameter is retuned when N changes.

Controller gains. All Energy–Casimir rows use shaping gain k_c=15, damping injection d_{inj}=2, and q-only drift-correction gain k_ξ=5.

Q-only sensing. The exact-port reference uses simulator velocity. The deployed rows use position measurements with Gaussian noise σ=0.01 and estimate velocity from recent position history.

Velocity estimation methods.

- **Oracle** uses the simulator velocity $\dot{q} = p/I_p$ and serves as the exact-port reference.
- **Finite difference** uses $\hat{\dot{q}}_t = (q_t - q_{t-1})/\Delta t$ and gives a simple causal q-only baseline.
- **MAP smoother** uses fixed-lag quadratic smoothing over the q-only window and gives a non-learned denoising baseline.
- **FD+TCN** applies a learned temporal correction to finite-difference features; we use it only in the pendulum q-only comparison.
- **C-PHAST observer** is the learned q-only front end used in the robot-arm transfer table.

The Energy–Casimir port interconnection schematic is shown in Figure 13 in the main text.

E.2. Stage 1: Single-Joint Validation

Stage 1 isolates the controller before any compositional transfer claim is made. The pendulum Hamiltonian is $H(q, p) = \frac{p^2}{2I_p} - mg\ell\cos(q)$, and the plant is integrated with RK4 (Δt=0.01, T=1000 steps).

Controllers compared.

- **Open-loop** ($u \equiv 0$) shows that the terminal-error metric is not trivial under the selected initial conditions.

- **PD** is a tuned conventional feedback baseline on the same plant.
- **Energy–Casimir (true)** uses the true port velocity and is the finite-step analogue of the exact-port calculation in Appendix A.
- **Energy–Casimir (PHAST)** replaces the true port velocity with the learned port-output estimate used by the q-only implementation.

Table 18. **Stage 1: Single-joint Energy–Casimir control (pendulum).** Entries are terminal equilibrium error after the $T{=}1000$ step rollout (mean $\pm$ std over 20 trials; lower is better). VelErr (PHAST) is the velocity-estimation MSE of the learned port-output observer used by the PHAST controller. Best terminal-error row per regime is in **bold**.

Regime	Open-loop	PD	Casimir (true)	Casimir (PHAST)	VelErr (PHAST)
Near stable	0.32 ± 0.21	$(4.51 \pm 3.20){\times}10^{-5}$	$(4.66 \pm 3.50){\times}10^{-5}$	$\mathbf{(2.20 \pm 1.70){\times}10^{-5}}$	$2.57{\times}10^{-3}$
Far stable	1.68 ± 0.71	$(2.45 \pm 0.86){\times}10^{-4}$	$(2.66 \pm 1.00){\times}10^{-4}$	$\mathbf{(1.29 \pm 0.53){\times}10^{-4}}$	$1.27{\times}10^{-2}$
Near unstable	3.45 ± 1.20	$(2.53 \pm 1.70){\times}10^{-5}$	$(2.29 \pm 1.50){\times}10^{-5}$	$\mathbf{(1.00 \pm 0.58){\times}10^{-5}}$	$1.38{\times}10^{-3}$
Far unstable	3.09 ± 0.17	$(1.29 \pm 0.084){\times}10^{-4}$	$(1.28 \pm 0.36){\times}10^{-4}$	$\mathbf{(4.86 \pm 0.64){\times}10^{-5}}$	$1.71{\times}10^{-2}$
High energy	3.19 ± 0.46	$(1.31 \pm 0.21){\times}10^{-4}$	$(1.92 \pm 0.72){\times}10^{-4}$	$\mathbf{(6.36 \pm 2.00){\times}10^{-5}}$	$2.14{\times}10^{-2}$

The PHAST-port row has lower terminal error than the true-velocity row in this finite-step experiment. This should be read as an implementation effect rather than a stronger controller theorem: the learned port output can behave like a smoothing estimate for the damping-injection channel.

Q-only velocity estimation. Table 19 separates the control law from the velocity-estimation problem under noisy position-only measurements. All rows stabilize these pendulum trials, but the final error, effort, and velocity-error columns show how much observer quality affects the deployed q-only controller.

Table 19. **Q-only velocity estimation comparison (pendulum).** Succ. is the fraction of trials that reach the equilibrium tolerance; final error is terminal position error; effort is cumulative squared torque; VelErr is velocity-estimation MSE against simulator velocity. Best non-oracle value per metric in **bold**.

Method	Succ.	Final error	Effort	VelErr
Casimir (true $\dot{q}$)	100.0%	0.00549 ± 0.0032	263	0
Casimir (finite diff.)	100.0%	0.0167 ± 0.01	342	1.12
Casimir (MAP smoother)	100.0%	$\mathbf{0.0117 \pm 0.0081}$	**325**	**0.133**
Casimir (FD+TCN)	100.0%	0.0132 ± 0.0093	336	0.169

E.3. Stage 2: Controller Attachment Across Tested Arm Counts

Stage 2 asks whether the same local controller remains usable after the plant is recomposed. We apply the identical per-joint Energy–Casimir gains from Stage 1 to planar-arm plants with $N \in \{2, 4, 8\}$ joints. The plant includes skew-symmetric inter-joint coupling ($\mathbf{C}_{\text{true}} = -\mathbf{C}_{\text{true}}^{\top}$, strength 1.0, normalized by $1/\sqrt{N}$).

Table 20. **Stage 2: Compositional Energy–Casimir control on robot arm.** Same per-joint gains as Stage 1, deployed on N-joint arms without retuning. Succ. is the fraction of trials reaching the task tolerance; Final RMS err. is terminal joint-space RMS error; effort is cumulative squared torque; VelErr is velocity-estimation MSE against simulator velocity. Mean $\pm$ std over 100 trials per regime. All rows succeed; among non-oracle estimators, best values for final error, effort, and VelErr are in **bold**.

N	Vel. Est.	Succ.	Final RMS err.	Effort	VelErr
			Near-equilibrium regime		
2	Oracle	100.0%	0.0049 ± 0.0021	15 ± 12	0.0000 ± 0.0000
2	FD	100.0%	0.0147 ± 0.0083	86 ± 12	0.5651 ± 0.0106
2	MAP	100.0%	0.0151 ± 0.0072	$\mathbf{60 \pm 11}$	$\mathbf{0.4502 \pm 0.0105}$
2	C-PHAST	100.0%	$\mathbf{0.0125 \pm 0.0069}$	87 ± 6	0.5825 ± 0.0099
4	Oracle	100.0%	0.0053 ± 0.0018	28 ± 16	0.0000 ± 0.0000
4	FD	100.0%	0.0160 ± 0.0049	170 ± 17	0.5647 ± 0.0087
4	MAP	100.0%	0.0165 ± 0.0049	$\mathbf{118 \pm 14}$	$\mathbf{0.4500 \pm 0.0072}$
4	C-PHAST	100.0%	$\mathbf{0.0156 \pm 0.0059}$	172 ± 12	0.5771 ± 0.0048
8	Oracle	100.0%	0.0050 ± 0.0013	54 ± 23	0.0000 ± 0.0000
8	FD	100.0%	$\mathbf{0.0159 \pm 0.0042}$	337 ± 23	0.5649 ± 0.0052
8	MAP	100.0%	0.0163 ± 0.0042	$\mathbf{233 \pm 20}$	$\mathbf{0.4499 \pm 0.0045}$
8	C-PHAST	100.0%	0.0180 ± 0.0056	341 ± 13	0.5761 ± 0.0049
			Moderate regime		
2	Oracle	100.0%	0.0048 ± 0.0023	57 ± 33	0.0000 ± 0.0000
2	FD	100.0%	0.0140 ± 0.0072	129 ± 32	0.5661 ± 0.0128
2	MAP	100.0%	0.0155 ± 0.0085	$\mathbf{99 \pm 30}$	$\mathbf{0.4508 \pm 0.0097}$
2	C-PHAST	100.0%	$\mathbf{0.0137 \pm 0.0071}$	131 ± 34	0.5781 ± 0.0121
4	Oracle	100.0%	0.0048 ± 0.0017	110 ± 44	0.0000 ± 0.0000
4	FD	100.0%	$\mathbf{0.0148 \pm 0.0060}$	254 ± 44	0.5657 ± 0.0067
4	MAP	100.0%	0.0157 ± 0.0057	$\mathbf{194 \pm 40}$	$\mathbf{0.4504 \pm 0.0067}$
4	C-PHAST	100.0%	0.0156 ± 0.0062	274 ± 45	0.5763 ± 0.0095
8	Oracle	100.0%	0.0049 ± 0.0011	239 ± 69	0.0000 ± 0.0000
8	FD	100.0%	$\mathbf{0.0157 \pm 0.0038}$	524 ± 68	0.5642 ± 0.0056
8	MAP	100.0%	0.0167 ± 0.0043	$\mathbf{406 \pm 63}$	$\mathbf{0.4516 \pm 0.0042}$
8	C-PHAST	100.0%	0.0160 ± 0.0033	543 ± 59	0.5774 ± 0.0076
			High-energy regime		
2	Oracle	100.0%	0.0044 ± 0.0022	228 ± 134	0.0000 ± 0.0000
2	FD	100.0%	0.0148 ± 0.0070	303 ± 131	0.5644 ± 0.0127
2	MAP	100.0%	$\mathbf{0.0143 \pm 0.0072}$	$\mathbf{258 \pm 121}$	$\mathbf{0.4515 \pm 0.0094}$
2	C-PHAST	100.0%	0.0148 ± 0.0064	300 ± 136	0.5773 ± 0.0104
4	Oracle	100.0%	0.0048 ± 0.0016	438 ± 178	0.0000 ± 0.0000
4	FD	100.0%	$\mathbf{0.0149 \pm 0.0049}$	588 ± 175	0.5661 ± 0.0091
4	MAP	100.0%	0.0170 ± 0.0067	$\mathbf{499 \pm 161}$	$\mathbf{0.4517 \pm 0.0063}$
4	C-PHAST	100.0%	0.0166 ± 0.0044	655 ± 181	0.5760 ± 0.0094
8	Oracle	100.0%	0.0052 ± 0.0012	953 ± 278	0.0000 ± 0.0000
8	FD	100.0%	0.0163 ± 0.0037	1249 ± 274	0.5668 ± 0.0050
8	MAP	100.0%	0.0162 ± 0.0036	$\mathbf{1069 \pm 254}$	$\mathbf{0.4532 \pm 0.0054}$
8	C-PHAST	100.0%	$\mathbf{0.0158 \pm 0.0041}$	1290 ± 242	0.5791 ± 0.0048

Discussion. The two-stage evaluation supports the paper's *control-compatibility* claim in a narrow, concrete sense: a pH controller block can attach through the same port variables preserved by C-PHAST, and the same per-joint storage-shaping gains stabilize the tested exact-model arm family $N \in \{2, 4, 8\}$, under the stated local assumptions, when copied without retuning. Under oracle velocity, the controller achieves near-perfect terminal accuracy for every N. Under q-only sensing, all tested observers stabilize these trials, while final error, effort, and velocity MSE reflect the observer-quality tradeoff predicted by Remark A.3.

F. Extended Method Details

F.1. Coordinate Maps, q-only Observation, and Canonicalization

This subsection expands the observation front end used by Algorithm 2. The pipeline has four maps: a deployment chart, a causal observer, a canonicalizer, and the compositional pH transition. The chart is the declared coordinate map that normalizes units, wraps angles, or handles simple coordinate constraints before measurements enter the model; the observer estimates the missing velocity or flow information from recent measurements; the canonicalizer converts that estimate into pH state variables; and the C-PHAST transition advances the typed subsystem states.

Measurement history and coordinate normalization. Let a deployment query end at time t, and write $a = t - L_h + 1$ for the first sample in the raw measurement window. For each index $\tau = a, \ldots, t$, let $\tilde{q}_\tau$ denote the raw measured coordinate

and

$$z_\tau = \chi_{\text{target}}(\tilde{q}_\tau)$$

denote the model-facing coordinate declared by the deployment specification. After this coordinate map, mechanical blocks write the local generalized coordinate as $q_\tau = z_\tau$ to match the PHASTCore notation. The coordinate-aware finite difference over this window uses

$$\dot{z}_a^{\text{fd}} = 0, \qquad \dot{z}_\tau^{\text{fd}} = \Delta_\chi(z_\tau, z_{\tau-1})/\Delta t, \quad \tau = a+1, \ldots, t, \tag{108}$$

where Δ_χ is ordinary subtraction in Euclidean or per-unit coordinates and the wrapped residual for angular coordinates.

Observer interface. Following PHAST's q-only formulation, we write the observer as a baseline smoother plus a residual:

$$\begin{aligned} \tilde{v}_{b:t} &= S_\eta(z_{b:t}), \\ \widehat{\dot{z}}_{b:t} &= \tilde{v}_{b:t} + r_\phi(z_{b:t}, \tilde{v}_{b:t}, \sigma_q), \end{aligned} \tag{109}$$

where $L_o \le L_h$ is the observer-window length and $b = t - L_o + 1$ is the first index in the observer window. Here S_η can be the finite-difference map in Eq. (108) restricted to $b{:}t$, a fixed-lag smoother, or another declared signal-processing baseline; r_ϕ is the learned causal residual that returns one correction per index in $b{:}t$; and σ_q is the measurement-noise scale when available. This factorization keeps the roles separate: the chart handles coordinate constraints, the smoother sets the physical signal scale, and the learned residual corrects the part needed by the pH core.

Noisy q-only and controller-facing port estimates. For noisy measurements, finite differences can amplify measurement noise as $O(\sigma_q/\Delta t)$. The PHAST implementation therefore also supports a MAP-first residual observer, which is the non-learned denoising baseline used in the q-only controller comparisons. For the same observer window $b{:}t$, first estimate a denoised model-coordinate trajectory

$$\begin{aligned} \hat{z}_{b:t}^{\text{MAP}} = \arg\min_{\zeta_b, \ldots, \zeta_t} \Bigg[& \sum_{j=b}^{t} \frac{\|\zeta_j - z_j\|^2}{\sigma_q^2} \\ & + \sum_{j=b+1}^{t-1} \frac{\|\zeta_{j+1} - 2\zeta_j + \zeta_{j-1}\|^2}{\sigma_a^2} \Bigg], \end{aligned} \tag{110}$$

then compute $\dot{z}_{b:t}^{\text{MAP}}$ from the smoothed trajectory and optionally add a learned residual:

$$\widehat{\dot{z}}_{b:t} = \dot{z}_{b:t}^{\text{MAP}} + r_\phi(z_{b:t}, \dot{z}_{b:t}^{\text{MAP}}, \sigma_q). \tag{111}$$

The MAP term removes high-frequency sensor noise before the learned residual is asked to estimate a port output.

Canonicalization. The canonicalizer maps the model-coordinate sample and estimated model-coordinate velocity into the latent pH state,

$$C_\psi : (z_t, \widehat{\dot{z}}_t) \mapsto (q_t, \hat{p}_t). \tag{112}$$

For the mechanical coordinates used in this paper, the only fact needed is that canonical momentum is mass-weighted velocity: $p = M(q)\dot{q}$. Thus the canonicalizer is the place where a velocity estimate becomes the momentum variable advanced by the pH dynamics. In the experiments in this paper the coordinate map is deliberately lightweight: identity, wrapped-angle, diagonal scaling, or per-unit electrical scaling, so the implemented canonicalizer is either $\hat{p} = \hat{\dot{q}}$ or a diagonal/effective-mass map $\hat{p} = M_{\text{diag}}\hat{\dot{q}}$ after identifying $q_t = z_t$. If a future deployment uses a nontrivial coordinate map $q = \varphi(z)$, the velocity would be pushed forward by $D\varphi(z)$ and the corresponding covector would transform by the cotangent lift; this is not needed for the reported benchmarks. In canonical coordinates (q, p), the mechanical pH structure has the standard block form

$$\begin{pmatrix} \dot{q} \\ \dot{p} \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix}}_{J_{\text{canonical}}} \nabla H(q,p) + \text{dissipation} + \text{control} \tag{113}$$

where the canonical skew block is independent of robot geometry.

Connection to Hamiltonian separability. The decoupling theory of (Ehrhardt et al., 2026) establishes that a monolithic pHODE can be split into subsystems only when a coordinate transformation makes the Hamiltonian separable: $H(x) = \sum_i H_i(w_i)$. Canonicalization moves the dynamics into canonical phase-space coordinates; when the kinetic energy is (approximately) separable across subsystems (e.g., diagonal or block-diagonal mass), this supports decomposition into shared per-subsystem blocks.

C-PHAST with canonicalization. After the deployment chart has mapped raw observations $\tilde{q}$ to model-facing coordinates z, the architecture writes the mechanical model coordinate as q inside the local pH block. It maps $(q, \hat{q})$ to canonical coordinates $(q, \hat{p})$, processes each subsystem with shared PHASTCore weights within type, applies skew-symmetric coupling, and decodes the model-coordinate readout back to observation space for output. Re-running the same chart, observer, and canonicalizer on a new measurement window is a state-refresh operation with fixed learned weights, not online parameter learning.

Q-only deployed rollout schematic. Figure 14 records the q-only observation-to-rollout wrapper behind Algorithm 2. We keep it in the appendix so the main method sequence can focus on the invariant contract, the local pH block, and the executable algorithms.

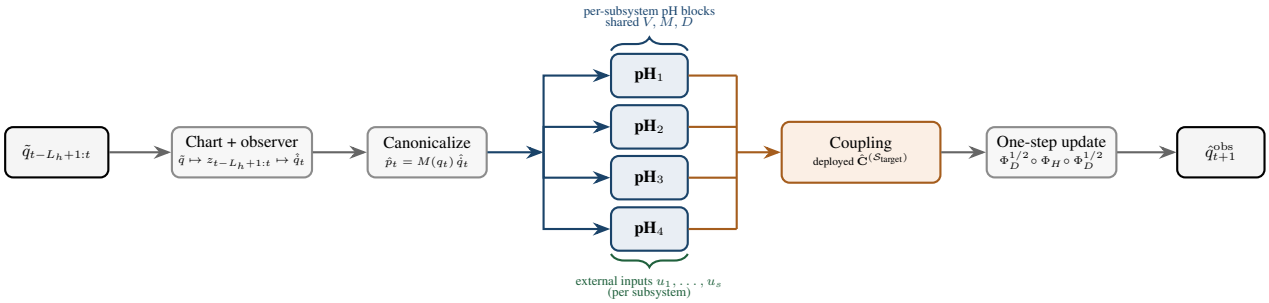

Figure 14. **C-PHAST deployed rollout pipeline.** From recent raw observation history, the deployment chart produces $z_{t-L_h+1:t}$ and the observer estimates velocity; canonicalization maps to phase space $(q_t, \hat{p}_t)$; blue marks the shared local port-Hamiltonian core; orange marks the deployed skew interconnection; and a light green side rail marks explicit per-subsystem external inputs. Together these components instantiate the deployed one-step transition operator $G_\theta^{(\mathcal{S}_{\text{target}})}$ used in Section 3. Algorithm 2 is the executable version of this diagram: its blue, orange, and green stage boxes correspond to the same local-core, deployed-coupling, and explicit-input roles shown here. This figure shows the deployed computation from q-only observations; the broader contract, deployment changes, and cross-domain composition patterns are given in Figures 1 and 5.

Monolithic vs. separated port-Hamiltonian views. Any interconnected collection of subsystems can be written as a single "monolithic" port-Hamiltonian system in the stacked state $x = (x_1, \ldots, x_s)$ with Hamiltonian H_{total} and interconnection matrix J_{coupled}. C-PHAST chooses the *separated* view because it exposes a low-dimensional interface between parts: each subsystem is integrated by its own PHASTCore step, while interactions occur only through port signals $(\hat{u}, \hat{y})$. This separation enables (i) parameter sharing within subsystem type, (ii) variable subsystem count or typed layout at test time by instantiating the coupling template for the target instance, and (iii) modular numerics.

F.2. PHASTCore Details

Each per-subsystem dynamics block in C-PHAST is a PHASTCore update: an explicit port-Hamiltonian step with learnable potential $V(q)$, mass $M(q)$, and damping $D(q)$ modules. The same block template supports three *knowledge regimes*: KNOWN instantiates structured physics modules, PARTIAL augments structured modules with learnable residuals, and LAW-UNKNOWN learns all components subject to physical constraints.

In the mechanical setting with state $x = (q, p)$, we use the mechanical Hamiltonian (separable only for constant/blockwise-constant mass; see (15))

$$H(q, p) = V(q) + \tfrac{1}{2} p^\top M(q)^{-1} p, \tag{114}$$

and choose a dissipation matrix that acts on momenta, $R = \operatorname{diag}(0, D(q))$ with $D(q) \succeq 0$.

Householder-style damping. Following PHAST, "Householder-style" means a low-rank additive PSD parameterization, not a Householder-reflection dynamics model. We parameterize configuration-dependent damping as a low-rank outer-product expansion:

$$D(q) = d_0 I + \sum_{i=1}^{r} \beta_i(q)\, k_i(q) k_i(q)^\top, \qquad d_0 \geq 0,\ \beta_i(q) \geq 0,\ \|k_i(q)\|_2 = 1, \tag{115}$$

This form guarantees $D(q) \succeq 0$ for all q while keeping matrix-vector products $O(nr)$. The essential requirement for the present paper is the PSD damping operator, because this is what preserves the pH energy-dissipation contract.

Bounding damping strength. The optional PHAST damping cap

$$\sum_{i=1}^{r} \beta_i(q) \leq \bar{\beta} \quad \Rightarrow \quad \lambda_{\max}(D(q)) \leq d_0 + \bar{\beta} \tag{116}$$

has two roles. First, it prevents damping from becoming an error sink for mistakes in V, M, the observer, or the input model. Second, with the Cayley damping step—unconditionally nonexpansive in the kinetic-energy metric—the cap is no longer required for basic finite-step stability, but it still improves accuracy in stiff regimes, linear-solve conditioning, and avoidance of the weakly damped sign-flipping that the method's lack of L-stability permits when $\Delta t\, \lambda_{\max}(D)$ is large. When the cap is active, the damping block is an attribution channel for irreversible loss rather than a generic numerical stabilizer.

Ordered modal coordinates. The unordered directions in Eq. (115) are sufficient for PSD damping and fast rollout, but they do not assign a stable identity to each dissipative direction. For diagnostics that need interpretable damping modes, PHAST can instead use an ordered partial-flag form

$$D_{\text{flag}}(q) = d_0 I + F(q)\, \mathrm{diag}(\alpha_1(q), \ldots, \alpha_r(q)) F(q)^\top, \qquad \alpha_1(q) \geq \cdots \geq \alpha_r(q) \geq 0, \tag{117}$$

with $F(q)^\top F(q) = I_r$. The nested spans of the first k columns of $F(q)$ define a partial flag. This is a coordinate system for the same PSD damping family, not a separate dynamics assumption. It is useful only when the dissipative law has a persistent hierarchy, because then a change in friction, contact loss, actuator damping, or electrical dissipation can be read as a change in ordered dissipative modes rather than as arbitrary motion of the conservative Hamiltonian core. Appendix J.2.1 uses this spectrum only as a conservative diagnostic channel in the Isaac push study.

Mass and the coupling metric. PHAST supports configuration-dependent $M(q) \succ 0$, but the C-PHAST benchmarks use either an identity/effective mass or a diagonal-mass canonicalizer unless a benchmark explicitly declares a richer mass model. The local Hamiltonian mass $M(q)$ should not be confused with the effective inverse-mass metric M_{eff}^{-1} used by the Cayley coupling step in Algorithm 1. The former belongs to the learned local kinetic energy; the latter is the deployment metric that converts skew exchange into a momentum update.

Single-block computation graph. Figure 15 records the single-block computation graph behind the shared PHASTCore update.

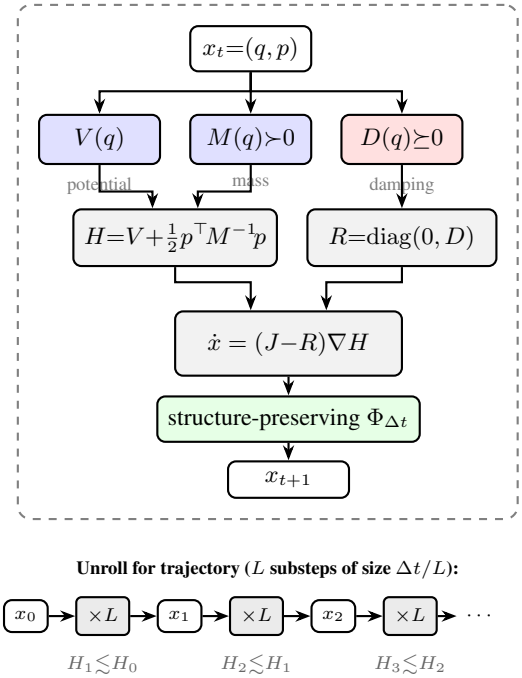

Figure 15. **PHASTCore: one step of port-Hamiltonian dynamics.** Components V, M, D (regime-dependent) assemble into Hamiltonian H and dissipation R; with closed external ports, the port-Hamiltonian form guarantees $dH/dt{\leq}0$ in continuous time. C-PHAST advances the phase state using a structure-preserving discrete-time map $\Phi_{\Delta t}$ and may compose L internal substeps per environment step. The $H_{k+1}{\lesssim}H_k$ annotations are schematic: finite-step energy behavior depends on timestep, integrator choice, and damping scale, which is why Eq. (116) is useful in stiff regimes.

F.3. Training Objective

This subsection makes explicit the losses summarized in Section 3. For a mini-batch of valid training windows $\mathcal{B}$, each pair $(i,t) \in \mathcal{B}$ indexes trajectory i and prediction time t, with history window $\tilde{q}^{(i)}_{t-L_h+1:t}$ available and target time $t+1$ inside the same trajectory. The model first applies the deployment coordinate map to the raw history window, infers a latent state $\hat{x}^{(i)}_t$ from the context ending at time t, advances it by one environment step of size Δt with the deployed C-PHAST transition, and decodes the model-coordinate readout to the observation-space prediction $\hat{q}^{\mathrm{obs},(i)}_{t+1}$. The observed target and input at that sample are $\tilde{q}^{(i)}_{t+1}$ and $u^{(i)}_t$. Let $\delta_q(a,b)$ denote the coordinate-aware prediction-error vector induced by the deployment chart: Euclidean coordinates use $a-b$, periodic coordinates use the wrapped difference in $(-\pi,\pi]$, and non-identity charts use the residual declared by the deployment chart. The data term is

$$\mathcal{L}_{\mathrm{data}} = \frac{1}{|\mathcal{B}|} \sum_{(i,t)\in\mathcal{B}} \left\| \delta_q\left(\tilde{q}^{(i)}_{t+1}, \hat{q}^{\mathrm{obs},(i)}_{t+1}\right) \right\|^2 . \tag{118}$$

For PHAST-family models, the optional physics terms use the predicted pH variables. For a sample (i,t), write

$$\Delta\hat{H}^{(i)}_t = H(\hat{x}^{(i)}_{t+1}) - H(\hat{x}^{(i)}_t), \qquad \hat{v}^{(i)}_t = \frac{\partial H}{\partial p}(\hat{x}^{(i)}_t),$$
$$\hat{y}^{(i)}_t = \hat{v}^{(i)}_t, \qquad \hat{P}^{\mathrm{ext},(i)}_t = (u^{(i)}_t)^\top \hat{y}^{(i)}_t,$$
$$\hat{P}^{\mathrm{diss},(i)}_t = -(\hat{v}^{(i)}_t)^\top D(\Pi_q(\hat{x}^{(i)}_t))\hat{v}^{(i)}_t.$$

For the mechanical benchmarks the input port output is velocity, $\hat{y}_t = \hat{v}_t$, and Π_q selects the model-coordinate configuration component entering the damping module. The powers are evaluated at the beginning of the one-step transition, so multiplying by Δt gives the left-endpoint discrete approximation to work over the step. For typed C-PHAST deployments, the same power quantities are computed per block/port and summed before entering the scalar loss. The skew interconnection

contributes zero net power by Theorem 3.1. The discrete passivity penalty is a one-sided surplus-energy hinge:

$$\mathcal{L}_{\text{pass}} = \frac{1}{|\mathcal{B}|} \sum_{(i,t)\in\mathcal{B}} \text{ReLU}\Big(\Delta \hat{H}_t^{(i)} - \Delta t\, \hat{P}_t^{\text{ext},(i)}\Big), \tag{119}$$

where the ReLU is applied to the scalar energy surplus inside the parentheses. Equivalently, it enforces the inequality $\Delta \hat{H}_t^{(i)} \le \Delta t\, \hat{P}_t^{\text{ext},(i)}$ up to hinge slack. Thus, closed-port predictions are penalized only when their Hamiltonian increases, and negative external port power penalizes trajectories that do not lose enough stored energy. The dissipated power appears in the stricter budget term rather than in the hinge. The energy-budget loss compares the predicted discrete Hamiltonian rate to the full pH power balance:

$$\mathcal{L}_{\text{energy}} = \frac{1}{|\mathcal{B}|} \sum_{(i,t)\in\mathcal{B}} \left| \frac{\Delta \hat{H}_t^{(i)}}{\Delta t} - \Big(\hat{P}_t^{\text{diss},(i)} + \hat{P}_t^{\text{ext},(i)}\Big) \right|. \tag{120}$$

When ground-truth dissipation-rate labels are supplied by a synthetic benchmark, we may replace the right-hand target in Eq. (120) with that label; the paper's robotics runs use the model-implied pH budget.

When reference velocities are available, the q-only observer receives direct supervision:

$$\mathcal{L}_{\text{observer}} = \frac{1}{|\mathcal{B}|} \sum_{(i,t)\in\mathcal{B}} \left\| \hat{\dot{q}}_t^{(i)} - \dot{q}_t^{(i)} \right\|^2. \tag{121}$$

Here $\hat{\dot{q}}_t^{(i)}$ is the observer-implied velocity estimate obtained from the inferred pH state, and $\dot{q}_t^{(i)}$ is the simulator reference velocity when that signal is available for supervision. More generally, when simulator phase or port variables are available during training, the same q-only deployed observer can receive typed auxiliary targets:

$$\begin{aligned} \mathcal{L}_p &= \|\hat{p}_t - p_t^\star\|^2, & \mathcal{L}_{\text{flow}} &= \left\| \widehat{\dot{q}}_t - M(q_t)^{-1} p_t^\star \right\|^2, \\ \mathcal{L}_{\text{port}} &= \left\| \hat{y}_t^{\text{port}} - \mathbf{B}^\top \nabla H(q_t, p_t^\star) \right\|^2. \end{aligned} \tag{122}$$

These are training-only targets for the observer/canonicalizer; when active they are folded into $\mathcal{L}_{\text{observer}}$ with the chosen weights. Deployment still begins from q-only histories through Eq. (109). The high-frequency observer term used only in Appendix I.2 is

$$\mathcal{L}_{\text{observer,hf}} = \frac{1}{|\mathcal{B}|} \sum_{(i,t)\in\mathcal{B}} \left\| \big(\hat{\dot{q}}_{t+1}^{(i)} - \hat{\dot{q}}_t^{(i)}\big) - \big(\dot{q}_{t+1}^{(i)} - \dot{q}_t^{(i)}\big) \right\|^2. \tag{123}$$

This term matches first differences of the inferred velocity, so it emphasizes rapid contact- or impact-scale changes that are muted by position-only supervision.

The optional rollout loss unrolls from a burn-in history of length L_h and compares the free-running future over training horizons $\mathcal{H}_{\text{train}}$:

$$\mathcal{L}_{\text{roll}} = \frac{1}{|\mathcal{B}|\,|\mathcal{H}_{\text{train}}|} \sum_{(i,t)\in\mathcal{B}} \sum_{h\in\mathcal{H}_{\text{train}}} \left\| \delta_q\Big(\tilde{q}_{t+h}^{(i)}, \hat{q}_{t+h|t}^{\text{obs},(i)}\Big) \right\|^2. \tag{124}$$

The notation $\hat{q}_{t+h|t}^{\text{obs},(i)}$ denotes the decoded free-running prediction at future offset h after initializing from the history ending at t. If rollout-physics training is enabled, Eqs. (119)–(120) are evaluated on the model's own rollout states rather than only on teacher-forced one-step states. The full objective is

$$\mathcal{L} = \lambda_{\text{data}}\mathcal{L}_{\text{data}} + \lambda_{\text{observer}}\mathcal{L}_{\text{observer}} + \lambda_{\text{observer,hf}}\mathcal{L}_{\text{observer,hf}} + \lambda_{\text{pass}}\mathcal{L}_{\text{pass}} + \lambda_{\text{energy}}\mathcal{L}_{\text{energy}} + \lambda_{\text{roll}}\mathcal{L}_{\text{roll}}. \tag{125}$$

All weights $\lambda_\cdot \ge 0$ are scalar hyperparameters; terms with zero weight are inactive. For matched-protocol comparisons we set $\lambda_{\text{data}} = 1$ and all auxiliary weights to zero unless otherwise stated; the full structured C-PHAST recipe activates the energy and observer terms where the required signals are available.

F.4. Shared Training Conventions

This subsection records the training choices that stay fixed unless a benchmark protocol explicitly overrides them. Readers should pair it with the benchmark-specific protocol blocks in Appendix H: the shared conventions say what stays fixed across domains, and the protocol blocks say what changes benchmark by benchmark.

Optimization. AdamW optimizer with learning rate 10^{-3}, weight decay 10^{-5}, cosine annealing over 50 epochs.

Loss function. All models use the next-step position term in Eq. (118). For C-PHAST and monolithic PHAST, the full structured recipe additionally uses the energy-budget term in Eq. (120) with λ_{energy}=1.0 and observer supervision in Eq. (121) with $\lambda_{\text{observer}}$=1.0 when simulator $\dot{q}$ is available. These auxiliary terms train internal pH variables and q-only observers, so they are not applied to non-pH baselines. The matched protocol trains all compared models for 50 epochs with only $\mathcal{L}_{\text{data}}$; the full structured PHAST-family rows are trained for 200 epochs with the active pH/observer terms stated in the relevant table caption.

Implementation. For reproducibility, all paper-grade results are generated on CPU (robot arm and MuJoCo benchmarks) or GPU (Isaac Lab: RTX 5090, 32 GB). A single C-PHAST training run (42,722 params, 200 epochs, Isaac Lab) takes $\sim$25 minutes; inference (rollout H=50 on the full test set) takes < 1 s. Fixed-dimension baselines are trained and evaluated at their fixed input dimension; C-PHAST is trained once on $N = 4$ and evaluated on all N. We report mean $\pm$ std over random seeds (typically 10 for the main C-PHAST variants and 3–5 for ablations/baselines).

G. Planar Arm Benchmark and Diagnostics

This appendix consolidates the planar-arm row of Table 6. The main result asks whether a model trained on a 4-joint arm can be re-instantiated at new joint counts. The schematic, protocol, companion one-step table, coupling ablations, energy checks, fixed-dimension baselines, and stale-URDF sanity check are kept together because they all explain the same repeated-joint count-transfer story.

G.1. Protocol and Schematic

This is the cleanest benchmark in the paper: it isolates repeated-joint count scaling without typed-layout, sparse-sensing, or embodiment complications. The robot arm is an N-joint planar manipulator with state $\boldsymbol{x} = (q, p) \in \mathbb{R}^{2N}$ where $q \in [-2.5, 2.5]^N$ are joint angles (with joint limits) and $p \in \mathbb{R}^N$ are momenta. Each joint has mass m_i=1, natural damping d_{nat}=0.1, and sinusoidal potential $V_i(q) = -\cos(q_i)$ (gravity). We simulate trajectories using semi-implicit Euler integration. Control inputs are per-joint sinusoids with random phases and frequencies, timestep $\Delta t = 0.02$s, trajectory length $T = 200$. Train: 200 trajectories, validate: 50, test: 50 per morphology $N \in \{2, 3, 4, 5, 6, 8\}$. The simulator injects inter-joint coupling through a fixed ground-truth skew matrix $\mathbf{C}_{\text{true}} \in \mathbb{R}^{N\times N}$: $u_{\text{coup}} = -\mathbf{C}_{\text{true}}\dot{q}$ with $\mathbf{C}_{\text{true}} = -\mathbf{C}_{\text{true}}^\top$. This term is power-preserving because $\dot{q}^\top u_{\text{coup}} = 0$. The learned model does not receive $\mathbf{C}_{\text{true}}$ in the LAW-UNKNOWN regime; it learns a deployed skew coupling $\hat{\mathbf{C}}$ instead. Figure 16 summarizes this repeated-joint simulator and the power-preserving coupling used to generate the data.

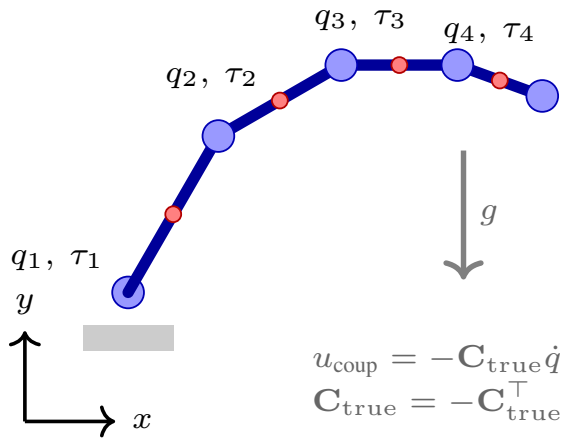

Figure 16. **Robot arm benchmark schematic.** An N-joint planar arm with joint-limited revolute joints $q \in [-2.5, 2.5]^N$, torques $u = \tau \in \mathbb{R}^N$, and q-only observations. Here $\mathbf{C}_{\text{true}}$ denotes the fixed simulator coupling used to generate data: $u_{\text{coup}} = -\mathbf{C}_{\text{true}}\dot{q}$ and $\mathbf{C}_{\text{true}} = -\mathbf{C}_{\text{true}}^\top$, so the coupling redistributes power across joints without injecting net energy.

G.2. Robot Arm Main-Result Companions

The main text keeps the rollout-stability table because it carries the variable-N claim. Table 21 is included here as companion evidence: it confirms that the same compositional models can be instantiated at all tested N, but it is less diagnostic than the rollout comparison in Table 12.

Table 21. Compositional generalization on robot-arm dynamics (q-only). Compositional models can be instantiated at unseen N, whereas fixed-dimension baselines cannot. One-step MSE. Train on $N{=}4$, test on all N. Mean $\pm$ std over seeds. Best per column in **bold**.

Model	Params	N=2	N=3	N=4	N=5	N=6	N=8
							MSE ($\times 10^{-6}$), mean $\pm$ std
C-PHAST (known coupling)	13k	**3.17**$\pm$0.05	2.00$\pm$0.01	1.70$\pm$0.00	2.49$\pm$0.03	1.90$\pm$0.01	1.95$\pm$0.01
C-PHAST (partial coupling)	22k	3.19$\pm$0.07	**1.96**$\pm$0.02	**1.66**$\pm$0.04	**2.47**$\pm$0.02	**1.86**$\pm$0.02	**1.92**$\pm$0.02
C-PHAST (learned skew, one-step)	23k	8.14$\pm$0.11	6.27$\pm$0.05	5.63$\pm$0.04	6.95$\pm$0.07	5.93$\pm$0.05	6.18$\pm$0.05
C-PHAST (learned skew, rollout-aligned)	23k	14.4$\pm$1.4	10.2$\pm$1.4	9.30$\pm$1.4	11.4$\pm$1.4	9.50$\pm$1.4	10.3$\pm$1.3

G.3. Coupling Parameterization

The coupling ablation asks which part of the compositional prior matters when the simulator contains the hidden skew inter-joint channel shown in Figure 16. All rows below are trained on the $N{=}4$ q-only arm and evaluated by instantiating the same learned template at each target joint count. The ablation compares three ways of handling the unknown inter-joint channel: no learned coupling, unconstrained neural coupling, and power-preserving skew coupling learned without access to $\mathbf{C}_{\mathrm{true}}$. One-step error mainly measures local fit; the $H{=}50$ takeover rollout is the more relevant stress test for whether the learned coupling remains stable after recomposition. The one-step ablation follows the expected local-fit pattern: Shared-GRU is best on this short-horizon metric, while the unknown-coupling C-PHAST variants are worse than the known/partial C-PHAST rows used for the main robot-arm result. We therefore tabulate the longer rollout stress below, where the limits of the unknown-coupling variants are most visible.

Table 22. **Robot-arm coupling ablation: 50-step takeover rollout MSE.** All C-PHAST variants here use the LAW-UNKNOWN regime, so this table isolates coupling parameterization rather than known-physics advantage. Shared-GRU's advantage reverses when correct physics priors are available in Table 12: C-PHAST KNOWN achieves 0.0014–0.0029, 3–6$\times$ better than Shared-GRU's 0.005–0.012. Mean $\pm$ std over seeds.

Model	Params	N=2	N=3	N=4	N=5	N=6	N=8
Shared-GRU	59,602	**0.0083 $\pm$ 0.0009**	**0.0050 $\pm$ 0.0007**	**0.0056 $\pm$ 0.0009**	**0.0057 $\pm$ 0.0005**	**0.0115 $\pm$ 0.0051**	**0.0079 $\pm$ 0.0015**
MPNN	42,178	0.5069 $\pm$ 0.2053	60.3547 $\pm$ 65.9677	0.5207 $\pm$ 0.0500	15.6683 $\pm$ 19.5141	3.0587 $\pm$ 1.8844	1.1246 $\pm$ 0.3670
C-PHAST (no coupling)	23,243	0.0744 $\pm$ 0.0212	0.0526 $\pm$ 0.0109	0.0505 $\pm$ 0.0119	0.0477 $\pm$ 0.0139	0.0533 $\pm$ 0.0136	0.0464 $\pm$ 0.0101
C-PHAST (neural coupling)	23,391	0.0649 $\pm$ 0.0208	0.0528 $\pm$ 0.0190	0.0510 $\pm$ 0.0157	0.0458 $\pm$ 0.0186	0.0508 $\pm$ 0.0164	0.0476 $\pm$ 0.0153
C-PHAST (skew coupling)	23,387	0.0721 $\pm$ 0.0226	0.0516 $\pm$ 0.0114	0.0494 $\pm$ 0.0126	0.0464 $\pm$ 0.0144	0.0523 $\pm$ 0.0140	0.0458 $\pm$ 0.0104

G.4. Energy Consistency

Rollout MSE alone can hide physically implausible trajectories when errors remain small in position space. On the same unknown-coupling rollouts, the per-joint Hamiltonian diagnostic follows the same ordering as the rollout table: Shared-GRU has the smallest numerical error, MPNN is unstable, and the C-PHAST unknown-coupling variants remain bounded but do not match the known/partial-physics rows in Table 12. We use this as a boundary check rather than as a separate headline table.

G.5. Robot Arm Hyperparameters

Table 23. Hyperparameters for C-PHAST robot arm experiments.

Parameter	Value
Potential hidden dim	64
Number of PHAST substeps per step (L)	1
Coupling type	Skew-symmetric
Coupling structure	Dense (learn max-size skew matrix; slice to N)
Coupling normalization	$1/\sqrt{N}$
Observer (q-only)	Finite-diff + TCN residual
Observer clamp	$\|\hat{\dot{q}}\|_\infty \leq 6$
Canonicalizer	Identity
Batch size	64
Learning rate	10^{-3}
Weight decay	10^{-5}
Epochs	50
Timestep Δt	0.02s
Trajectory length	200
History length L_h	5
Rollout horizon H	50

G.6. Fixed-Dimension Baselines

Table 24 contrasts fixed-dimension baselines, including non-compositional PHAST regimes, with C-PHAST at N=4. "—" indicates structural impossibility from shape mismatch, not poor performance.

Table 24. Fixed-dimension baselines vs C-PHAST on robot-arm dynamics (q-only). One-step MSE at N=4. Fixed-dimension methods cannot be evaluated at $N \neq 4$ due to hard-coded input projections. Mean ± std over seeds. Best per column in **bold**.

Model	Params	MSE (N=4)	MSE (N=6)	MSE (N=8)	Generalizes?
Fixed-dimension baselines (trained at N=4; not compositional)					
PHAST (known coupling)	4,039	$(1.660\pm0.007)\times10^{-6}$	—	—	✗
PHAST (partial coupling)	14,547	$(1.670\pm0.002)\times10^{-6}$	—	—	✗
PHAST (learned skew)	19,433	$(1.58\pm0.02)\times10^{-6}$	—	—	✗
S5	17,732	0.0047 ± 0.0003	—	—	✗
LRU	34,372	0.0088 ± 0.0005	—	—	✗
LinOSS	17,732	0.0449 ± 0.0028	—	—	✗
GRU	39,428	0.0083 ± 0.0002	—	—	✗
Compositional architecture (per-joint weight sharing)					
C-PHAST (partial coupling)	21,977	$(1.66\pm0.04)\times10^{-6}$	$(1.86\pm0.02)\times10^{-6}$	$(1.92\pm0.02)\times10^{-6}$	✓

G.7. Coupling Diagnostics

Beyond prediction error, we verify that the learned coupling behaves as a *power-preserving* interconnection. For skew-symmetric coupling, the coupling power residual $(y^{\text{port}})^\top \hat{u}$ should be numerically zero (Theorem 3.1).

Table 25. **Coupling diagnostics on robot-arm dynamics (q-only).** The coupling power residual $(y^{\text{port}})^\top \hat{u}$ should vanish for a skew interconnection (Theorem 3.1). The table reports mean and maximum absolute residuals over held-out trajectories and the relative skew-symmetry defect of $\hat{\mathbf{C}}$; values near machine precision verify that the learned interconnection preserves power rather than injecting energy.

N	$\mathbb{E}[\lvert(y^{\text{port}})^\top \hat{u}\rvert]$	$\max \lvert(y^{\text{port}})^\top \hat{u}\rvert$	$\lVert\hat{\mathbf{C}} + \hat{\mathbf{C}}^\top\rVert_F / \lVert\hat{\mathbf{C}}\rVert_F$
2	3.14×10^{-9}	5.96×10^{-8}	0
3	5.16×10^{-9}	1.24×10^{-7}	0
4	7.23×10^{-9}	1.19×10^{-7}	0
5	5.48×10^{-9}	9.00×10^{-8}	0
6	5.76×10^{-9}	1.02×10^{-7}	0
8	5.28×10^{-9}	9.00×10^{-8}	0

G.8. Coupling Normalization Ablation

We also checked whether the learned skew-coupling channel should be deployed with no normalization, $1/\sqrt{N}$, or $1/N$ scaling. In this unknown-coupling diagnostic, prediction metrics at $N{=}8$ are unchanged to the reported precision: all three variants give one-step MSE 0.0741 ± 0.0054 and $H{=}50$ rollout MSE $\approx 2.364 \pm 0.101$. The normalization mainly changes the spectral norm and numerical power residual of the learned coupling, as expected, so we use $1/\sqrt{N}$ as the default scaling rather than treating this sweep as a separate empirical claim.

G.9. Configuration-Dependent Coupling

Table 26 compares constant coupling $\hat{\mathbf{C}}$ with configuration-dependent coupling $\hat{\mathbf{C}}(q)$. The configuration-dependent variant achieves 2–3$\times$ lower takeover rollout error for $N \leq 4$ (interpolation), while constant coupling matches or outperforms for $N > 4$ (extrapolation).

Table 26. Configuration-dependent coupling $\hat{\mathbf{C}}(q)$ vs. constant $\hat{\mathbf{C}}$ on robot-arm dynamics (q-only, KNOWN regime). Takeover rollout MSE at $H{=}50$. Train on $N{=}4$. Best per column in **bold**.

	Takeover Rollout MSE ($H{=}50$, $\times10^{-3}$)					
N	2	3	4	5	6	8
$\hat{\mathbf{C}}$ (constant)	2.53±0.03	1.98±0.02	2.51±0.04	**1.81±0.02**	**1.74±0.02**	**1.74±0.03**
$\hat{\mathbf{C}}(q)$ (config-dep.)	**0.77±0.08**	**1.51±0.10**	**1.05±0.08**	2.83±0.21	1.78±0.08	2.14±0.10

G.10. Controlled Mass-Prior Corruption (Toy Stale-URDF Check)

Table 27 is intentionally a toy sanity check rather than the paper's main embodiment-mismatch claim. It perturbs only the planar-arm mass prior while keeping topology, sensing, and actuation fixed, so it should be read as a controlled misspecification probe for the KNOWN/PARTIAL regimes, not as a realistic fixed-layout sim-to-real workflow. Under this limited corruption, KNOWN/PARTIAL degrade gracefully ($\sim$1.2$\times$ at $\epsilon{=}20\%$, $\sim$3.6$\times$ at $\epsilon{=}50\%$), and even at 50% mass error the physics-informed models still outperform LAW-UNKNOWN by $\sim$30$\times$.

Table 27. Controlled mass-prior corruption on robot-arm dynamics (q-only). Takeover rollout MSE at $H{=}50$ under mass perturbation ϵ. Best per row in **bold**.

	KNOWN		**PARTIAL**		**UNKNOWN**	
ϵ	$N{=}4$	$N{=}8$	$N{=}4$	$N{=}8$	$N{=}4$	$N{=}8$
0.00	$(2.46\pm0.04)\times10^{-3}$	$(1.65\pm0.03)\times10^{-3}$	$(2.48\pm0.06)\times10^{-3}$	$(1.66\pm0.05)\times10^{-3}$	$(2.87\pm0.04)\times10^{-1}$	$(2.61\pm0.03)\times10^{-1}$
0.05	$(2.26\pm0.02)\times10^{-3}$	$(1.47\pm0.01)\times10^{-3}$	$(2.26\pm0.02)\times10^{-3}$	$(1.47\pm0.02)\times10^{-3}$	$(2.87\pm0.04)\times10^{-1}$	$(2.61\pm0.03)\times10^{-1}$
0.10	$(2.24\pm0.03)\times10^{-3}$	$(1.45\pm0.01)\times10^{-3}$	$(2.26\pm0.03)\times10^{-3}$	$(1.46\pm0.02)\times10^{-3}$	$(2.87\pm0.04)\times10^{-1}$	$(2.61\pm0.03)\times10^{-1}$
0.20	$(2.92\pm0.06)\times10^{-3}$	$(2.08\pm0.04)\times10^{-3}$	$(2.94\pm0.06)\times10^{-3}$	$(2.09\pm0.05)\times10^{-3}$	$(2.87\pm0.04)\times10^{-1}$	$(2.61\pm0.03)\times10^{-1}$
0.50	$(8.77\pm0.20)\times10^{-3}$	$(7.45\pm0.18)\times10^{-3}$	$(8.78\pm0.19)\times10^{-3}$	$(7.45\pm0.16)\times10^{-3}$	$(2.87\pm0.04)\times10^{-1}$	$(2.61\pm0.03)\times10^{-1}$

H. Remaining Benchmark Protocols and Reading Guide

This section mirrors the non-planar benchmark order of Table 6. The planar-arm protocol and diagnostics are consolidated in Appendix G, because the hidden skew-coupling simulator, schematic, and ablations are easiest to read as one block. Each

subsection below records the protocol-level facts needed to reproduce the remaining benchmarks; later appendices return to the same benchmark only when additional diagnostics are needed. If the main text gave the invariant contract, this section is the place to see the contract instantiated concretely enough to rerun the benchmark without reverse-engineering hidden assumptions.

H.1. Legged Robot Benchmark Details (MuJoCo)

This is the appendix block for the homologous repeated-leg story: same leg template, different leg count, before the fixed-layout embodiment shift introduced later by Isaac Lab.

We use MuJoCo-based quadruped, biped, and hexapod with consistent per-leg joint structure. Each leg is treated as a subsystem with local state $(q_{\text{leg}}, p_{\text{leg}})$; legs are coupled via skew-symmetric $\hat{\mathbf{C}}$. Observations are q-only (joint encoders); velocity is estimated via the TCN observer. This appendix benchmark is a homologous count-transfer family, not the Isaac Lab cross-robot benchmark: all morphologies reuse the same 2-DOF leg template and differ only in the number of repeated leg blocks. Train on quadruped (4 legs), test zero-shot on biped (2 legs) and hexapod (6 legs). The one-step diagnostic is less informative than the rollout table in the main text: all C-PHAST coupling variants agree to three decimal places, while Shared-GRU fits the training morphology slightly better but diverges when recomposed to unseen leg counts. We therefore keep the main-text rollout table as the quantitative result and use Figure 17 only to make the repeated-leg transfer visually inspectable.

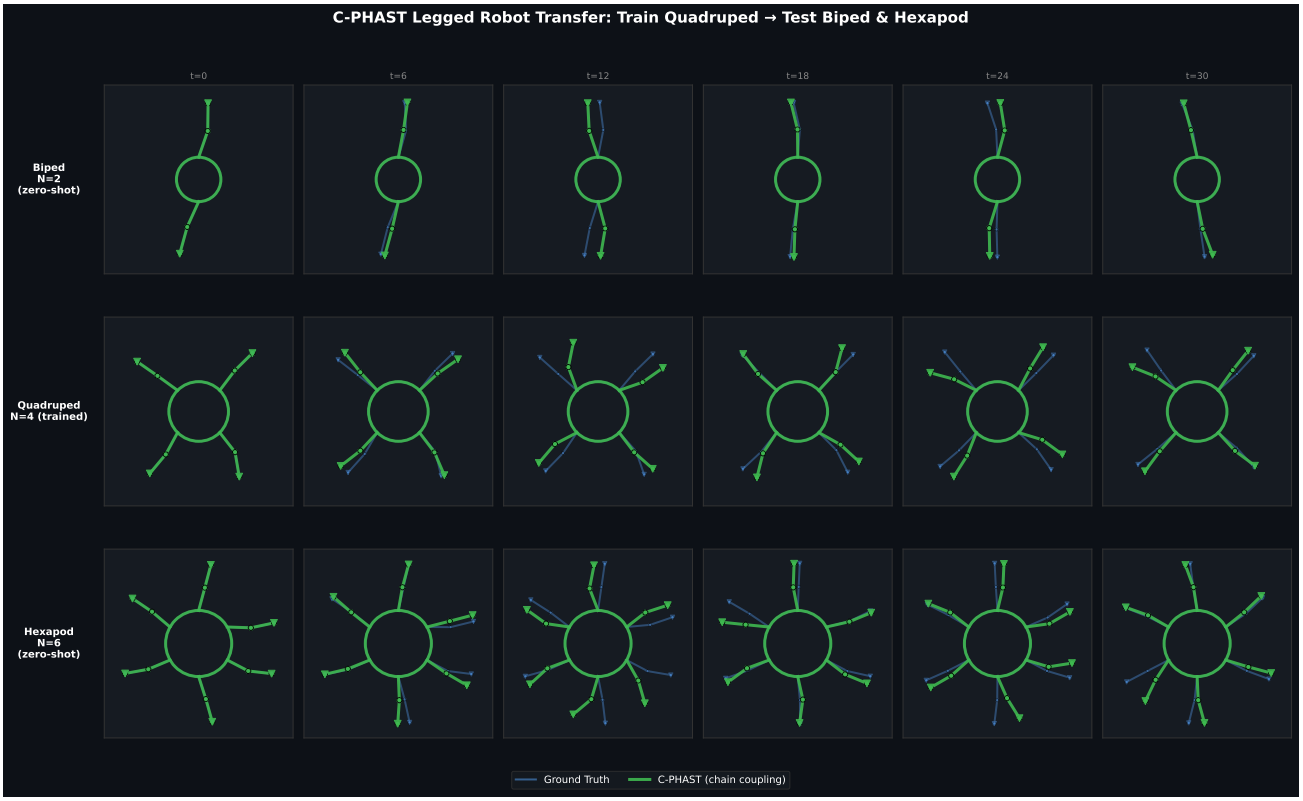

Figure 17. **MuJoCo homologous leg-count transfer filmstrip.** C-PHAST is trained on the quadruped (N=4 repeated leg blocks) and re-instantiated without target retraining on held-out biped (N=2) and hexapod (N=6) systems that share the same 2-DOF per-leg joint type. Blue traces are ground truth and green traces are C-PHAST rollouts. This figure is qualitative; Table 13 gives the quantitative rollout comparison against baselines.

H.2. Coupled FitzHugh–Nagumo Benchmark Details

An FHN unit is a compact two-state excitable circuit: the voltage-like state can spike, the recovery state pulls it back, and a diffusive link lets neighboring voltages influence one another. The benchmark uses that simple circuit family to test repeated-block recomposition outside robotics: copy the same excitable-cell block, then rebuild the chain graph for a

new cell count. This appendix block gives the protocol behind Table 14; its scope is coupled excitable circuits rather than cardiac-field or bidomain readout modeling.

Single-cell dynamics. We use the standard FitzHugh–Nagumo dynamics

$$\dot{V} = V - \frac{V^3}{3} - W + I, \qquad \dot{W} = \epsilon(V + a - bW),$$

with the benchmark parameters used in the codebase: $\epsilon = 0.08$, $a = 0.7$, $b = 0.8$, and default stimulus $I = 1.0$ for the single-cell generator unless otherwise specified by the coupled-data routine. The state is (V, W), where V is the voltage-like coordinate and W is the recovery variable. C-PHAST represents each unit as one shared excitable-cell pH block; KNOWN fixes the analytical FHN terms, PARTIAL keeps the pH scaffold while learning the nonlinear constitutive component, and LAW-UNKNOWN learns the block from data under the same training recipe as the other fully learned rows.

Port-Hamiltonian realization and energy ledger. Taking the (capacitive plus inductive) storage $H(V, W) = \frac{1}{2}V^2 + \frac{1}{2\epsilon}W^2$ (with C=1, L=$1/\epsilon$), the single-cell dynamics above yield the exact power balance

$$\dot{H} \;=\; \underbrace{\left(V^2 - \tfrac{1}{3}V^4\right)}_{\text{tunnel diode } R_1 \text{ (active)}} \;+\; \underbrace{V\,I}_{\text{stimulus port}} \;+\; \underbrace{a\,W}_{\text{source } E=-a} \;-\; \underbrace{b\,W^2}_{R_2 \text{ (passive, } \geq 0)}, \tag{126}$$

which we verified against the simulator to machine precision. The linear resistor $R_2 = b$ (matching the $-bW^2$ term above, with W the inductor current and $L = 1/\epsilon$) is the only passive term; the cubic tunnel diode R_1 contributes $V^2(1 - \frac{1}{3}V^2)$, which is *positive* (energy-injecting) for $|V| < \sqrt{3}$ and negative otherwise. Each cell is therefore an *active* element: it carries an active constitutive port rather than a positive-semidefinite dissipation matrix, and so does not lie in the passive $\dot{x} = (J - R)\nabla H$ class; the stimulus I and source $E = -a$ enter as external ports. Cells couple diffusively through the membrane voltage, whose network power is $\sum_i V_i \sum_j g_{ij}(V_j - V_i) = -\frac{1}{2}\sum_{ij} g_{ij}(V_i - V_j)^2 \leq 0$: this interconnect is passive and *dissipative*, unlike the skew, zero-net-power coupling of the mechanical and microgrid blocks (Theorem 3.1). The benchmark accordingly tests repeated-block *recomposition*, which is independent of whether the interconnect is skew or dissipative; (126) makes the block's active/passive accounting auditable rather than assumed.

Coupled-chain protocol. For the coupled benchmark, N identical units are connected in a chain with diffusive voltage coupling strength $g = 0.1$. The transfer axis is therefore concrete: the cell equations and coupling law stay fixed, while the number of excitable cells and the chain Laplacian change at deployment. Training uses N=3, 200 trajectories, 200 time steps, and $\Delta t = 0.1$. At test time the same shared unit block is copied to $N \in \{2, 3, 4, 5, 6, 8\}$ and the chain Laplacian is rebuilt for the target count. The N-PHDAE baseline is trained as a single-cell model and composed over the same known chain graph. Reported values are means over seeds 42–44 using 100 training epochs.

Interpretation. The exact KNOWN C-PHAST rows are reference rows for integration and analytical-port correctness. The learned comparison is more conservative: N-PHDAE is the stronger learned raw forecaster at fixed N=3, while C-PHAST PARTIAL and LAW-UNKNOWN demonstrate that the shared-block architecture recomposes without a size-dependent error increase. This is why the main text uses FHN as a bridge between mechanical repeated-block transfer and heterogeneous microgrid composition, not as a blanket superiority claim over N-PHDAE.

H.3. Isaac Lab Benchmark Details

This is the fixed-layout deployment-mismatch benchmark: same four-leg block layout, but different embodiment, contact regime, and local mechanical response across commercial robots. Within the three-way deployment taxonomy used in the main text, this appendix block is primarily about *response mismatch*: the observation contract is already restricted to q-only sensing, while the low-level controller interface is held fixed rather than studied as a separate deployment axis.

Robots. We use ANYmal-D (ANYbotics) and Unitree Go2, which are commercially distinct quadrupeds with different mass distributions, link lengths, and actuator properties. Both are simulated in NVIDIA Isaac Lab (IsaacSim 5.1) at 50 Hz control frequency (decimation 4 from 200 Hz physics). Each robot has N=12 actuated joints organized as s=4 legs with d=3 joints per leg: hip abduction/adduction (HAA), hip flexion/extension (HFE), and knee flexion/extension (KFE).

Policy training. Locomotion policies are trained using RSL-RL on flat terrain with the default reward function (linear velocity tracking, angular velocity tracking, joint torque penalties). Training runs for $\sim$2000 iterations ($\sim$10M environment steps) per robot. Policies achieve stable trotting gaits at $\sim$1 m/s.

Data collection. We record joint angles q_t and torques τ_t from the trained policies deployed in Isaac Lab. The matched cross-robot split used in Table 8 trains on 350 ANYmal-D trajectories, validates on 50 ANYmal-D trajectories, and evaluates zero-shot on 100 Go2 trajectories, each with $T{=}200$ steps. Action noise is applied during data collection (Gaussian, $\sigma{=}0.05$ rad) to diversify the training distribution. No velocity or contact information is recorded; models observe position-only.

Cross-robot protocol. C-PHAST is trained on ANYmal-D data only and evaluated zero-shot on Go2 (no target-domain training data). Unlike the MuJoCo benchmark above, this experiment keeps the block layout fixed at 4 legs $\times$ 3 joints per leg and changes only the robot-specific body parameters and contact regime. Because both robots expose the same 12-actuated-joint layout, fixed-dimension baselines are applicable here; unlike the variable-N benchmarks, this is not a shape-mismatch test. Rows in the matched transfer comparison are trained on ANYmal-D and tested on Go2 without target retraining.

H.4. DC Microgrid Benchmark Details

A DC microgrid is a direct-current electrical network in which converter-controlled buses feed loads through transmission lines. In this benchmark the buses are Distributed Generation Units (DGUs), the cables are line blocks, and the operational questions are familiar power-system ones: can bus voltage stay regulated, can current sharing remain balanced, and what changes when the feeder graph or available sensors change? This subsection supports the DC-microgrid row of Table 6, the main-text regime summary in Table 15, and the full appendix results in Tables 28, 29, 30, and 31. It is the benchmark where the typed-block story is most explicit: readers come here to see exactly how the DGU/line decomposition, topology rewiring, and voltage-only deployment tier are instantiated.

Typed blocks and nominal graph family. The benchmark consists of N Distributed Generation Units (DGUs) connected by M transmission lines. Each DGU is modeled as a 1-DOF pH subsystem with charge $q{=}CV$ and flux $p{=}LI_t$, driven by Buck-converter voltage u_i and load current I_{Li}. Each transmission line is a separate 1-DOF pH subsystem with inductive line dynamics. The two typed blocks are coupled through the oriented incidence matrix B_{inc}: DGU charge equations receive line currents via $B_{\text{inc}} I_{\text{line}}$, while line momentum equations receive voltage differences via $-B_{\text{inc}}^{\top} V$. This is a node-edge interconnection rather than the chain-style repeated-block coupling used in the robotics benchmarks.

Nominal training protocol. We use the Neary benchmark parameters (Neary et al., 2024): $R_t{=}1.2\,\Omega$, $L_t{=}1.8\,\text{H}$, $C_t{=}2.2\,\text{F}$, with heterogeneous line parameters sampled from $\mathcal{U}(0.1, 2.0)$. Time-varying converter inputs $u_i(t)$ and load currents $I_{Li}(t)$ are passed through the compositional action pathway. Training uses chain topologies at $N{=}3$ (100 trajectories, 100 steps, $\Delta t{=}0.01$) and transfers to $N \in \{2, 3, 4, 5, 8\}$ by reusing the shared DGU and line cores while rebuilding the incidence coupling for each target graph. N-PHDAE follows the same compose-and-predict protocol with known graph structure.

Topology-reconfiguration tier. The reconfiguration benchmark keeps typed blocks fixed but rewires the deployment graph at test time. We train on chain layouts at $N{=}4$, then evaluate on ring and star rewires while still providing the new topology to the model. Operationally, this is a stylized proxy for feeder reconfiguration after a protection or overload event: one switch pattern is deactivated, another tie path is activated, and the same DGU and line assets must remain usable under the new incidence pattern. This tier is where the microgrid benchmark most directly mirrors the three-arm toy example in Section 3: $B_{\text{inc}} \to B'_{\text{inc}}$ rebuilds the deployed interconnection while the learned typed local cores remain fixed.

Voltage-only partial-sensing tier. The hardest deployment regime keeps topology explicit but restricts deployment observations to bus voltages. This mirrors the practical case where voltage sensors are available at the buses but line currents and internal converter states are not exposed directly to the deployment model. This is the appendix location to look for the sparse-sensing version of the benchmark: hidden-state reconstruction becomes part of the task, and the design-facing metrics (voltage-balancing error, current-sharing residual, and $|\Delta H|$ error) become as important as raw rollout MSE.

Microgrid diagnostic metrics. The microgrid tables use the same benchmark diagnostics throughout. $H{=}50$ denotes observed rollout MSE over a 50-step rollout. Vbal err is the RMS error in the Cucuzzella-style weighted-average voltage-balancing residual (Cucuzzella et al., 2019). The maintained benchmark has no per-DGU capacity ratings, so the reported runs use equal weights and the residual reduces to arithmetic-average bus-voltage balancing against the nominal voltage references. Share err is the RMS error in the proportional-current-sharing residual $WI_t - \overline{WI_t}\mathbf{1}$, with the same equal-weight convention in the reported benchmark. $|\Delta H|$ err is absolute error in the Hamiltonian energy-change diagnostic. Hidden MSE is reported only in partial-sensing runs, where line currents and converter states are latent. Energy-throughput efficiency, local voltage-deviation, and current coefficient-of-variation diagnostics remain in the API output for auditability, but they are not used as the paper's voltage-regulation or current-sharing metrics. Lower is better for all reported errors.

Typed residual metrics. The microgrid rerun also records the residual channels induced by the decomposition in Eq. (13). Observation residuals measure bus-voltage mismatch; DGU-current and line-current residuals measure typed local-state mismatch; energy-balance excess measures violation of the one-step storage balance after accounting for dissipated and supplied port work; port-work residuals compare local converter/load work; and interconnect-edge-power residuals compare edge-level power exchange. For deployment diagnostics we report the reference-normalized amplification $A_c(s) = r_c(s)/(r_c(\mathrm{ref}) + \epsilon)$, because the reference seed variance is numerically tiny and would make a z-score misleading. Table 29 gives the raw C-PHAST PARTIAL residuals behind the main-text amplification table.

Full numeric microgrid tables. The main text compresses these results into a regime-summary table. We keep the full numeric tables here so the boundary claim remains auditable: N-PHDAE is stronger in the full-state known-graph rollout regime, while C-PHAST PARTIAL becomes most useful when the deployment stress is topology reconfiguration or design-facing sparse-sensing metrics. Table 28 gives the full-state known-graph composition result, Table 29 gives the raw residual channels behind the main-text amplification table, Table 30 gives topology rewiring, and Table 31 gives the voltage-only deployment tier.

Table 28. **DC Microgrid: heterogeneous typed-block composition and transfer.** One typed-block pH template composes DGUs and transmission lines across graph sizes. This table reports the one-step part of the full-state known-graph benchmark: C-PHAST PARTIAL attains the best error at the training size $N{=}3$, while N-PHDAE is strongest at the held-out graph sizes in this view. Train on $N{=}3$ DGUs in a chain and transfer to $N \in \{2, 3, 4, 5, 8\}$. Time-varying $u_i(t)$ and $I_{Li}(t)$ are supplied through action ports. Mean $\pm$ std over 3 seeds, 100 epochs. OLS denotes the per-type ordinary-least-squares warm start from PHAST (Bhardwaj & Bajaj, 2026). Best per column is shown in **bold green**.

	1-Step MSE ↓				
Model	$N{=}2$	$N{=}3$ (train)	$N{=}4$	$N{=}5$	$N{=}8$
C-PHAST PARTIAL	$2.4{\times}10^{-4}$	$\mathbf{2.4{\times}10^{-7}}$	$3.7{\times}10^{-5}$	$9.4{\times}10^{-5}$	$6.0{\times}10^{-5}$
C-PHAST KNOWN	$2.2{\times}10^{-4}$	$1.4{\times}10^{-6}$	$3.1{\times}10^{-5}$	$8.5{\times}10^{-5}$	$5.5{\times}10^{-5}$
N-PHDAE (composed)	$\mathbf{3.5{\times}10^{-6}}$	$3.3{\times}10^{-6}$	$\mathbf{3.2{\times}10^{-6}}$	$\mathbf{3.2{\times}10^{-6}}$	$\mathbf{3.1{\times}10^{-6}}$
C-PHAST LAW-UNKNOWN+OLS	$3.5{\times}10^{-5}$	$2.0{\times}10^{-4}$	$7.0{\times}10^{-5}$	$2.6{\times}10^{-5}$	$2.9{\times}10^{-4}$

Table 29. **Raw typed residuals for C-PHAST PARTIAL under microgrid reconfiguration.** These are the raw residual means whose reference-normalized amplifications appear in Table 16. The diagnostic pattern is channel-specific: DGU removal and ring rewiring raise energy, line-current, and interconnect-power channels, while star rewire stays near the reference scale.

Scenario	Voltage	Line current	Energy excess	Port work	Interconnect power
Reference chain	$8.49{\times}10^{-4}$	$6.13{\times}10^{-3}$	$1.66{\times}10^{-4}$	$1.45{\times}10^{-2}$	$1.93{\times}10^{-2}$
Removed DGU	$1.89{\times}10^{-2}$	$2.56{\times}10^{-1}$	$3.04{\times}10^{-2}$	$1.66{\times}10^{-2}$	$8.35{\times}10^{-1}$
Added DGU	$3.13{\times}10^{-3}$	$3.30{\times}10^{-2}$	$4.29{\times}10^{-3}$	$1.75{\times}10^{-2}$	$7.78{\times}10^{-2}$
Ring rewire	$5.82{\times}10^{-3}$	$4.26{\times}10^{-2}$	$6.21{\times}10^{-3}$	$1.46{\times}10^{-2}$	$1.14{\times}10^{-1}$
Star rewire	$7.19{\times}10^{-4}$	$5.76{\times}10^{-3}$	$1.56{\times}10^{-4}$	$1.45{\times}10^{-2}$	$1.67{\times}10^{-2}$

Table 30. **Topology-aware microgrid reconfiguration.** Train on chain layouts at $N{=}4$, then evaluate on rewired graphs while still providing the new topology. $H{=}50$ is observed rollout MSE; Vbal err is the Cucuzzella-style voltage-balancing error; $|\Delta H|$ err is Hamiltonian energy-change error; and share err is proportional-current-sharing residual error. N-PHDAE remains the strongest raw predictor on the harder ring rewrite, but C-PHAST PARTIAL is the tighter design surrogate relative to both N-PHDAE and the LAW-UNKNOWN+OLS warm-start variant. The KNOWN row is included as a near-analytical reference. Best per scenario/metric is shown in **bold green**.

Scenario	Model	$H{=}50\downarrow$	Vbal err $\downarrow$	$\lvert\Delta H\rvert$ err $\downarrow$	share err $\downarrow$
Ring rewire	C-PHAST KNOWN	$3.17{\times}10^{-3}$	$4.19{\times}10^{-5}$	$1.24{\times}10^{-1}$	$2.07{\times}10^{-4}$
	C-PHAST PARTIAL	$3.79{\times}10^{-3}$	$5.48{\times}10^{-5}$	$1.49{\times}10^{-1}$	$3.26{\times}10^{-4}$
	C-PHAST LAW-UNKNOWN+OLS	$1.40{\times}10^{-2}$	$1.48{\times}10^{-1}$	$6.70{\times}10^{1}$	$6.21{\times}10^{-3}$
	N-PHDAE	$5.82{\times}10^{-5}$	$1.67{\times}10^{-3}$	$6.52{\times}10^{-1}$	$2.99{\times}10^{-3}$
Star rewire	C-PHAST KNOWN	$2.77{\times}10^{-4}$	$4.31{\times}10^{-5}$	$3.70{\times}10^{-2}$	$8.29{\times}10^{-5}$
	C-PHAST PARTIAL	$4.00{\times}10^{-5}$	$5.73{\times}10^{-5}$	$2.52{\times}10^{-2}$	$1.22{\times}10^{-4}$
	C-PHAST LAW-UNKNOWN+OLS	$2.71{\times}10^{-2}$	$1.49{\times}10^{-1}$	$6.73{\times}10^{1}$	$6.59{\times}10^{-3}$
	N-PHDAE	$6.48{\times}10^{-5}$	$1.62{\times}10^{-3}$	$5.30{\times}10^{-1}$	$3.40{\times}10^{-3}$

Table 31. **Voltage-only partial sensing on the microgrid.** Topology is still provided, but only bus voltages are observed at deployment time. Obs 1-step and $H{=}50$ obs are computed only on observed bus-voltage channels; hidden MSE evaluates reconstruction of latent currents/states. In the 5-seed rerun, C-PHAST PARTIAL attains the best one-step and hidden-state errors, C-PHAST KNOWN attains the lowest long-horizon observed-rollout error, and N-PHDAE is narrowly best on voltage balancing and $|\Delta H|$. Best per column is shown in **bold green**.

Model	obs 1-step $\downarrow$	hidden MSE $\downarrow$	$H{=}50$ obs $\downarrow$	Vbal err $\downarrow$	$\lvert\Delta H\rvert$ err $\downarrow$
C-PHAST PARTIAL	$4.34{\times}10^{-5}$	$4.78{\times}10^{-1}$	$3.92{\times}10^{-2}$	$6.68{\times}10^{-2}$	$2.36{\times}10^{1}$
C-PHAST KNOWN	$4.58{\times}10^{-5}$	$5.03{\times}10^{-1}$	$3.83{\times}10^{-2}$	$6.88{\times}10^{-2}$	$2.43{\times}10^{1}$
N-PHDAE	$4.50{\times}10^{-5}$	$4.89{\times}10^{-1}$	$3.92{\times}10^{-2}$	$6.60{\times}10^{-2}$	$2.34{\times}10^{1}$
C-PHAST LAW-UNKNOWN+OLS	$8.51{\times}10^{-5}$	$4.91{\times}10^{-1}$	$7.38{\times}10^{-2}$	$8.57{\times}10^{-2}$	$3.95{\times}10^{1}$

Full sensing-degradation ladder. Table 32 extends the main-paper voltage-only point into a four-tier observation ladder: full state, bus voltages plus DGU currents, bus voltages plus one line current, and bus voltages only. The purpose is not to create a new headline benchmark, but to verify that the sparse-sensing story degrades in the expected order and that the design-facing ranking does not invert unexpectedly as information is removed. Figure 18 provides the same result as a compact visual trend summary.

Table 32. **Microgrid sensing-degradation ladder.** Matched 5-seed rerun at $N{=}4$ chain topology, reported as means across observation tiers from full state to voltages only. Removing current channels increases rollout and Vbal errors for the three stable models once line-current information is unavailable. C-PHAST KNOWN, C-PHAST PARTIAL, and N-PHDAE remain close under partial sensing, while the LAW-UNKNOWN+OLS warm-start variant has much larger design errors and worse sparse-rollout behavior. Here $V + I_{\mathrm{DGU}}$ denotes bus voltages plus local DGU currents, $V + I_{\mathrm{line}}$ denotes bus voltages plus a single line-current channel, and the metric columns use the definitions above. Best per sensing tier/metric is shown in **bold green**; the full-state hidden-MSE zeros are left unhighlighted because no state is hidden.

Sensing tier	Model	obs 1-step ↓	hidden MSE ↓	$H{=}50$ obs ↓	Vbal err ↓	$\lvert\Delta H\rvert$ err ↓
Full state	C-PHAST KNOWN	$2.91{\times}10^{-7}$	0.00	$8.61{\times}10^{-5}$	$1.82{\times}10^{-5}$	$4.71{\times}10^{-2}$
	C-PHAST PARTIAL	$5.64{\times}10^{-8}$	0.00	$2.11{\times}10^{-5}$	$1.69{\times}10^{-5}$	$3.99{\times}10^{-2}$
	C-PHAST UNKNOWN+OLS	$7.55{\times}10^{-5}$	0.00	$4.09{\times}10^{-2}$	$1.62{\times}10^{-1}$	$7.36{\times}10^{1}$
	N-PHDAE	$3.23{\times}10^{-6}$	0.00	$6.58{\times}10^{-5}$	$1.56{\times}10^{-3}$	$5.11{\times}10^{-1}$
$V + I_{\mathrm{DGU}}$	C-PHAST KNOWN	$4.09{\times}10^{-6}$	$2.25{\times}10^{-1}$	$1.18{\times}10^{-3}$	$1.90{\times}10^{-5}$	$2.96{\times}10^{-1}$
	C-PHAST PARTIAL	$4.25{\times}10^{-6}$	$2.34{\times}10^{-1}$	$1.42{\times}10^{-3}$	$1.61{\times}10^{-5}$	$3.42{\times}10^{-1}$
	C-PHAST UNKNOWN+OLS	$1.95{\times}10^{-5}$	$2.33{\times}10^{-1}$	$2.66{\times}10^{-2}$	$1.62{\times}10^{-1}$	$7.34{\times}10^{1}$
	N-PHDAE	$8.65{\times}10^{-6}$	$2.33{\times}10^{-1}$	$1.18{\times}10^{-3}$	$1.56{\times}10^{-3}$	$7.12{\times}10^{-1}$
$V + I_{\mathrm{line}}$	C-PHAST KNOWN	$2.69{\times}10^{-5}$	$4.49{\times}10^{-1}$	$1.11{\times}10^{-2}$	$3.61{\times}10^{-2}$	$1.17{\times}10^{1}$
	C-PHAST PARTIAL	$2.66{\times}10^{-5}$	$4.40{\times}10^{-1}$	$9.10{\times}10^{-3}$	$3.41{\times}10^{-2}$	$1.08{\times}10^{1}$
	C-PHAST UNKNOWN+OLS	$1.92{\times}10^{-4}$	$4.53{\times}10^{-1}$	$8.76{\times}10^{-2}$	$1.35{\times}10^{-1}$	$6.13{\times}10^{1}$
	N-PHDAE	$2.63{\times}10^{-5}$	$4.26{\times}10^{-1}$	$9.46{\times}10^{-3}$	$3.72{\times}10^{-2}$	$1.18{\times}10^{1}$
Voltages only	C-PHAST KNOWN	$4.58{\times}10^{-5}$	$5.03{\times}10^{-1}$	$3.83{\times}10^{-2}$	$6.88{\times}10^{-2}$	$2.43{\times}10^{1}$
	C-PHAST PARTIAL	$4.34{\times}10^{-5}$	$4.78{\times}10^{-1}$	$3.92{\times}10^{-2}$	$6.68{\times}10^{-2}$	$2.36{\times}10^{1}$
	C-PHAST UNKNOWN+OLS	$8.51{\times}10^{-5}$	$4.91{\times}10^{-1}$	$7.38{\times}10^{-2}$	$8.57{\times}10^{-2}$	$3.95{\times}10^{1}$
	N-PHDAE	$4.50{\times}10^{-5}$	$4.89{\times}10^{-1}$	$3.92{\times}10^{-2}$	$6.60{\times}10^{-2}$	$2.34{\times}10^{1}$

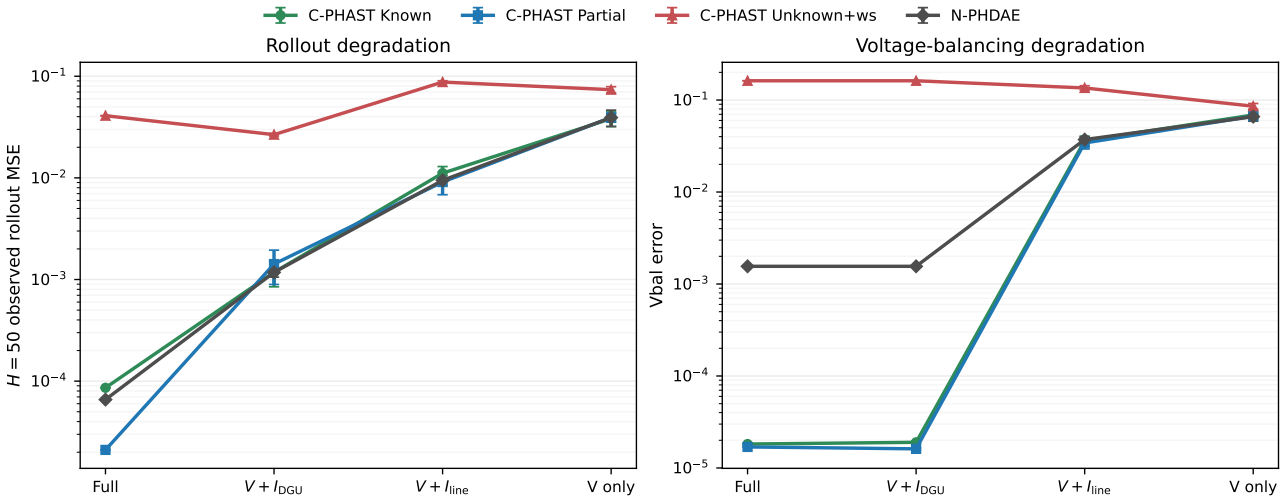

Figure 18. **Microgrid sensing-degradation trends.** Left: $H{=}50$ observed rollout MSE. Right: Cucuzzella-style voltage-balancing error. The three stable models remain close across sensing tiers; removing line-current information raises both observed-rollout and voltage-balancing errors, while the LAW-UNKNOWN+OLS warm-start variant is already poor on the voltage-balancing metric and remains worse on sparse-rollout errors. Error bars show $\pm 1\sigma$ over 5 seeds.

I. Legged-Robot Transfer Diagnostics and Data Budget

This appendix is the legged-robot companion to Section 4.2. The fixed-layout Isaac Lab transfer experiment trains on ANYmal-D locomotion data collected under a trained RL policy in PhysX, then evaluates on Go2 without retraining. All diagnostics below use the held-out test set (100 trajectories, 100 timesteps each at $\Delta t = 0.02\,\mathrm{s}$). Within the legged-robot row of Table 6, this block explains both the behavior of the zero-shot transfer result and the target-data budget needed to beat it.

The evidence chain has four parts. First, jerk statistics test whether the q-only rollout interface smooths contact-rich simulator motion. Second, the observer ablation asks whether forcing sharper high-frequency estimates improves source accuracy at the cost of target transfer. Third, the few-shot study asks how much Go2 data is needed when a source-trained model is adapted. Fourth, the from-scratch scaling study asks how much target data a flexible baseline needs before it overtakes the structural prior.

I.1. Jerk Analysis: Quantifying the Smoothness Prior

To directly quantify the $\mathcal{C}^2$ smoothness prior imposed by Hamiltonian flow, we measure **jerk** ($\dddot{q}$, the third time-derivative of joint position) across ground truth, C-PHAST, and Shared-GRU predictions. Jerk is the natural diagnostic: smooth Hamiltonian dynamics produce trajectories with bounded jerk, while contact events create jerk spikes from acceleration discontinuities.

Ground truth confirms $\mathcal{C}^{-1}$ contact dynamics. Table 33 reports jerk statistics on the held-out test set (100 trajectories, 100 timesteps, Δt=0.02 s). Ground-truth jerk is high (mean 3,392 rad/s^3 on ANYmal-D, 4,153 on Go2) and heavy-tailed. Defining "contact events" as timesteps with jerk magnitude above the 95th percentile, these events exhibit $4.25\times$ higher mean jerk than non-contact timesteps (12,406 vs. 2,918), confirming genuine $\mathcal{C}^{-1}$ discontinuities in the training data. Per-joint-type analysis shows the knee flexion/extension (KFE) joint has the highest jerk (3,560), consistent with direct ground contact.

Teacher-forced predictions: C-PHAST amplifies jerk. In teacher-forced mode, each model prediction is independently initialized from the ground-truth state at time t. C-PHAST's teacher-forced jerk is $2.5\times$ GT (mean 8,562), while Shared-GRU closely matches GT ($1.08\times$). This counter-intuitive result arises because teacher-forced predictions are *not* connected trajectories: each step independently initializes a Hamiltonian integration from a new $(q_t, \hat{p}_t)$, where $\hat{p}_t$ is estimated by the observer. Small variations in the observer's momentum estimate across consecutive timesteps produce phase-shifted integrations that, when differenced three times, amplify into high jerk. The smoothness prior of Hamiltonian flow applies to *connected* trajectories, not to sequences of independent 1-step predictions.

Rollout predictions under q-only sensing: smoothness prior dominates. Under autoregressive rollout (20-step context, 80 predicted steps), both models produce dramatically smoother trajectories than ground truth. C-PHAST's rollout jerk is 0.03% of GT (mean 1.04), while Shared-GRU retains 1.0% (mean 32.8), giving a $30\times$ gap between models. Neither model produces realistic gait dynamics autoregressively; both converge toward near-stationary predictions. However, C-PHAST's Hamiltonian flow imposes the strongest smoothness constraint, making its rollout trajectories $30\times$ smoother than the unconstrained Shared-GRU. This is consistent with the "bounded-error" interpretation of rollout stability under the restricted q-only interface used here: without explicit contact/support ports, unobserved impact impulses are treated as unexplained high-frequency variation, so the pH prior favors bounded smooth continuations rather than fitting every contact transient. This should not be read as a claim that C-PHAST cannot model contact-rich dynamics; the deployment contract must expose the relevant contact/support information through typed ports if those events are to be predicted rather than smoothed.

Table 33. **Jerk diagnostic for q-only contact smoothing ($|\dddot{q}|$, rad/s^3).** Teacher-forced (TF) initializes each one-step prediction from the ground-truth state; rollout (RO) is autoregressive from a 20-step context. Ratio is mean jerk divided by the corresponding ground-truth mean for the same robot. Colored cells mark the diagnostic extremes used in the text, not task-performance winners.

Robot	Source	Mean jerk	Ratio / diagnostic
ANYmal-D	GT	3,392	1.00×
	C-PHAST (TF)	8,562	2.52×
	Shared-GRU (TF)	3,665	1.08×
	C-PHAST (RO)	1.0	0.0003×
	Shared-GRU (RO)	32.8	0.010×
Go2	GT	4,153	1.00×
	C-PHAST (TF)	9,927	2.39×
	Shared-GRU (TF)	4,531	1.09×
	C-PHAST (RO)	1.4	0.0003×
	Shared-GRU (RO)	34.0	0.008×
Contact analysis (ANYmal-D GT; events are >95th percentile jerk):			
	Contact events	12,406	4.25×
	Non-contact	2,918	

I.2. Observer Ablation Under q-only Sensing

The jerk diagnostic above shows that, under the q-only Isaac transfer interface used in this paper, the learned rollout is smoother than the contact-rich simulator trajectory. This is a statement about the observation contract, not a limitation of port-Hamiltonian modeling in general: if contact state, support forces, foot velocities, or ground-reaction impulses are exposed as typed ports, the model has an explicit channel through which contact-rich dynamics can enter. The ablation below asks a narrower question: can the q-only observer alone recover enough high-frequency contact information from joint histories to close the source accuracy gap without sacrificing transfer? We systematically test four strategies, holding all other hyperparameters fixed at the v9 configuration (hidden dim $= 64$, n_{layers}=2, per-joint-type Hamiltonians, λ_{energy}=1.0, $\lambda_{\text{observer}}$=1.0, 200 epochs).[3]

Table 34. **Observer ablation on Isaac Lab cross-robot transfer (train ANYmal-D, test Go2).** AD denotes ANYmal-D source validation and Go2 denotes the zero-shot target. RF is the observer receptive field in timesteps; ctx is rollout context length; λ_{roll} weights Eq. (124); λ_{hf} weights the high-frequency observer term in Eq. (123). Baseline and best config (S4, λ_{hf}=5.0) are validated with 5 seeds (mean±std shown); other configs are single-seed. Best source-side one-step and rollout accuracies are shown in **bold green**; transfer columns are left unhighlighted because they encode the tradeoff.

Strategy	Configuration	1-Step (AD)	H=50 (AD)	H=50 (Go2)	Transfer Gap
Baseline	RF=7, per-joint	5.98×10^{-4}	0.057±0.001	0.065±0.001	1.15±0.02×
S1: Receptive field expansion (larger temporal context)					
	RF=13 (kernel 5, 2 layers)	5.57×10^{-4}	0.058	0.067	1.16×
	RF=29 (kernel 5, 3 layers)	5.57×10^{-4}	0.058	0.068	1.19×
S2: Multi-joint observer (per-leg grouped processing)					
	Per-leg (n_{dof}=3), RF=7	5.65×10^{-4}	0.056	0.068	1.23×
	Per-leg (n_{dof}=3), RF=29	5.50×10^{-4}	0.058	0.066	1.14×
S3: Multi-step rollout loss (rollout-aware backprop through pH integrator)					
	H=5, λ_{roll}=0.1, ctx=5	5.98×10^{-4}	0.058	0.065	1.13×
	H=5, λ_{roll}=0.5, ctx=5	6.01×10^{-4}	0.058	0.065	1.13×
S4: High-frequency supervision (temporal-difference matching)					
	λ_{hf}=1.0	5.96×10^{-4}	0.058	0.070	1.21×
	λ_{hf}=2.0	5.99×10^{-4}	0.055	0.068	1.23×
	λ_{hf}=5.0	6.03×10^{-4}	0.052±0.004	0.066±0.002	1.27±0.08×

[3]This ablation uses per-joint-type Hamiltonians (42,722 params) with stronger observer supervision ($\lambda_{\text{observer}}$=1.0). The main table uses shared Hamiltonians (22,862 params) with the same energy and observer protocol. All comparisons within this ablation are against a consistent internal baseline.

Architecture changes improve one-step but not rollout (S1–S2). Expanding the receptive field from 7 to 29 timesteps ($0.14 \rightarrow 0.58$ s, covering a full gait cycle) improves one-step MSE by 7% but leaves rollout $H{=}50$ unchanged. Similarly, switching from independent per-joint processing to grouped per-leg processing (3 DOFs jointly) improves one-step MSE by 6–8% with no rollout benefit. Both strategies slightly worsen the transfer gap (from $1.13\times$ to 1.14–$1.23\times$), consistent with the hypothesis that richer observer representations capture robot-specific patterns.

Multi-step rollout loss has no effect (S3). Training with an autoregressive rollout loss (backpropagating through the pH integrator for $H{=}5$ or $H{=}10$ steps) produces *identical* test metrics despite being the dominant gradient signal during training (total training loss $\sim$1.27 vs. $\sim$0.001 for data MSE alone). The observer is already co-adapted to the integrator: the rollout loss reinforces essentially the same optimum that one-step supervision already finds.

High-frequency supervision reveals the fundamental tradeoff (S4). Adding the high-frequency observer term from Eq. (123) with weight λ_{hf} penalizes the observer's tendency to smooth high-frequency contact transients. This is the *only* intervention that substantially improves rollout accuracy: at $\lambda_{\text{hf}}{=}5.0$, source rollout drops from 0.057 to **0.052** (-8%, 5 seeds: $p{<}0.05$), the largest improvement across all strategies. However, the transfer gap monotonically widens: $1.15\times \rightarrow 1.21\times \rightarrow 1.23\times \rightarrow 1.27\times$ as λ_{hf} increases from 0 to 5.

Interpretation. These results establish that, under q-only sensing, the observer's smoothing behavior is the mechanism behind the accuracy–transfer tradeoff, not merely a correlation. Strategies S1–S3 show that the smoothing is not a capacity or training limitation; the observer has sufficient capacity and gradient signal to produce sharper velocity estimates. Strategy S4 confirms that when we *force* sharper estimates via high-frequency supervision, source accuracy improves but transfer degrades because the q-only observer must infer contact timing and ground-reaction impulses indirectly, and those inferred high-frequency patterns are morphology-specific. The smooth velocity estimates produced by the baseline observer act as an implicit *domain-invariant representation*: by discarding robot-specific high-frequency content, they retain only the low-frequency gait structure that generalizes across morphologies. The operational implication is constructive: to improve contact-rich forecasting without giving up transfer, the next model should not ask the observer to hallucinate all contact physics from q alone; it should expose support/contact variables as deployment ports and let the pH block account for their power exchange explicitly.

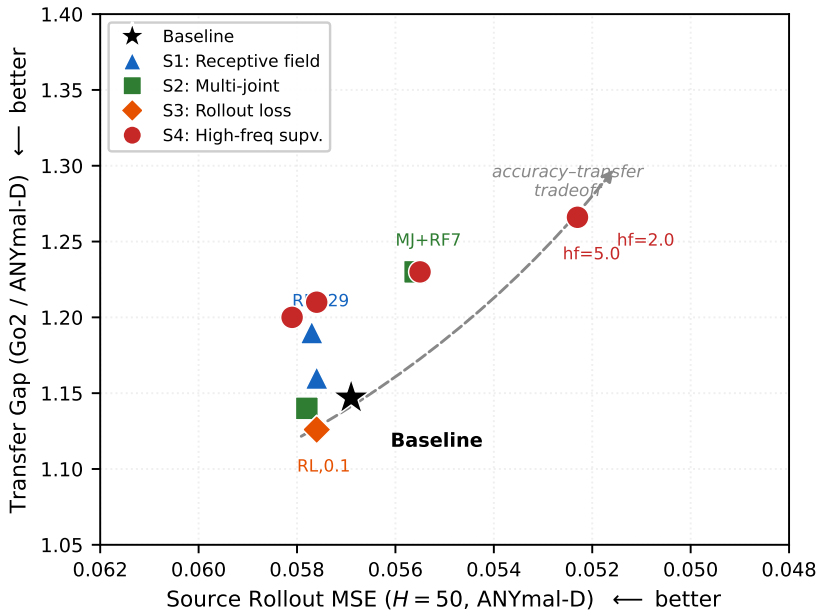

Figure 19. Accuracy–transfer Pareto plot for observer ablation (Table 34). Each point is a training configuration; ideal is bottom-right (low source error, low transfer gap). Strategies S1–S3 cluster near the baseline with no rollout improvement. Only S4 (high-frequency supervision, red circles) moves along the tradeoff frontier: better source accuracy at the cost of worse transfer.

I.3. Few-Shot Target Adaptation

Section 4.4 presents the headline *one-step* result: C-PHAST's zero-shot Go2 error (9.8×10^{-4}) outperforms Shared-GRU adapted with 50 target trajectories (17.0×10^{-4}) by $1.7\times$. This subsection provides the full experimental details, extended one-step results across all data regimes and freezing strategies, and analysis of the structural floor–ceiling duality. It answers the practical question raised by the legged-robot row: how much target data is needed before adaptation beats transfer?

I.3.1. Experimental Setup

Pre-trained models. We reuse the same source-trained checkpoints as the matched-protocol Isaac transfer experiment in Section 4.2: both models are trained on 350 ANYmal-D trajectories for 50 epochs with one-step MSE only. C-PHAST uses the clean baseline architecture (shared Hamiltonian, $n_{\text{dof\_per\_leg}}=3$) with 22,862 parameters. Shared-GRU uses 60,804 parameters. We report 5 seeds (42–46).

Adaptation data. From the 200 Go2 trajectories collected under the trained RSL-RL policy, we allocate the first N for adaptation training and 100 held-out trajectories for testing. We sweep $N \in \{5, 10, 25, 50\}$.

Adaptation protocol. Fine-tuning uses AdamW with $\text{lr}=10^{-4}$ ($10\times$ lower than from-scratch training) for 20 epochs. Batch size is $\min(N, 32)$ to accommodate small-N regimes.

Freezing strategies. We compare two strategies:

- **Full adaptation**: all parameters trainable.
- **Core-preserving adaptation**: Hamiltonian and integration blocks frozen; only observer, coupling, and canonicalizer parameters are updated. For Shared-GRU, the core GRU weights are frozen.

Evaluation. After adaptation, we evaluate on: (1) Go2 test set (100 trajectories), measuring adaptation quality; (2) ANYmal-D test set (100 trajectories), measuring catastrophic forgetting on the source domain. We report one-step MSE ($\times 10^{-4}$) only; rollout adaptation is not evaluated in this subsection.

I.3.2. Results

Table 11 in the main text summarizes the full one-step fine-tune results. Table 35 below extends this with core-frozen results.

Table 35. Extended few-shot *one-step* adaptation results (mean ± std over 5 seeds). One-step MSE ($\times 10^{-4}$); no rollout metric is reported here. **Target**: Go2 test MSE after adaptation. **Source**: ANYmal-D test MSE (forgetting check). N=0 is the zero-shot baseline. ↓ indicates lower is better. Best Go2 target value within each model family and best source-retention value are shown in **bold green**.

Model	Freeze	N	Go2 (target) ↓	AD (source) ↓
C-PHAST (22,862 params)				
	—	0	9.8±0.1	4.8±0.0
	Full	5	9.6±0.1	4.8±0.0
	Core	5	9.6±0.1	4.8±0.0
	Full	10	9.6±0.1	4.8±0.0
	Core	10	9.6±0.1	4.8±0.0
	Full	25	9.6±0.1	4.8±0.0
	Core	25	9.6±0.1	4.8±0.0
	Full	50	9.5±0.1	4.8±0.0
	Core	50	9.5±0.1	4.8±0.0
Shared-GRU (60,804 params)				
	—	0	25.5±1.0	4.4±0.4
	Full	5	20.0±1.2	7.8±1.2
	Core	5	22.0±1.0	6.1±0.9
	Full	10	19.1±1.2	7.9±1.2
	Core	10	21.3±1.1	5.9±0.9
	Full	25	19.1±1.3	8.1±1.2
	Core	25	21.3±1.1	6.0±0.9
	Full	50	17.0±1.3	8.2±1.3
	Core	50	19.5±1.3	7.0±1.1

I.3.3. ANALYSIS

The extended table reinforces the main-text one-step takeaway without changing the conclusion. C-PHAST improves only marginally with target data ($9.8 \rightarrow 9.5 \times 10^{-4}$ from zero-shot to N=50), and full fine-tuning matches the core-frozen setting to reported precision at every budget. That plateau is consistent with a structural floor–ceiling picture: the transferable physics prior is already doing most of the work, while the remaining robot-specific mismatch lies outside the model's adaptation directions.

Shared-GRU moves much more under adaptation ($25.5 \rightarrow 17.0 \times 10^{-4}$ at N=50), but it pays for that plasticity with substantial forgetting on the source robot ($4.4 \rightarrow 8.2 \times 10^{-4}$). Core-freezing partially reduces that forgetting while also limiting target improvement. Even at the largest budget in this sweep, Shared-GRU's best adapted Go2 error remains $1.7\times$ worse than C-PHAST's zero-shot error, so the practical conclusion is unchanged: the structural prior is already worth more than 50 target trajectories in this one-step adaptation regime.

I.4. Target-Data Scaling from Scratch

A complementary question to few-shot *adaptation* (Appendix I.3) is from-scratch *training*: how many target-domain trajectories does a GRU need before it surpasses C-PHAST's zero-shot transfer? This is the scaling companion to the legged-robot row of Table 6: it estimates how many target trajectories the structural prior is worth.

We train both C-PHAST and Shared-GRU from scratch on $N \in \{5, 10, 25, 50, 100, 350\}$ Go2 trajectories (no ANYmal-D pretraining), evaluating rollout MSE at H=50 on 100 held-out Go2 trajectories across 3 seeds.

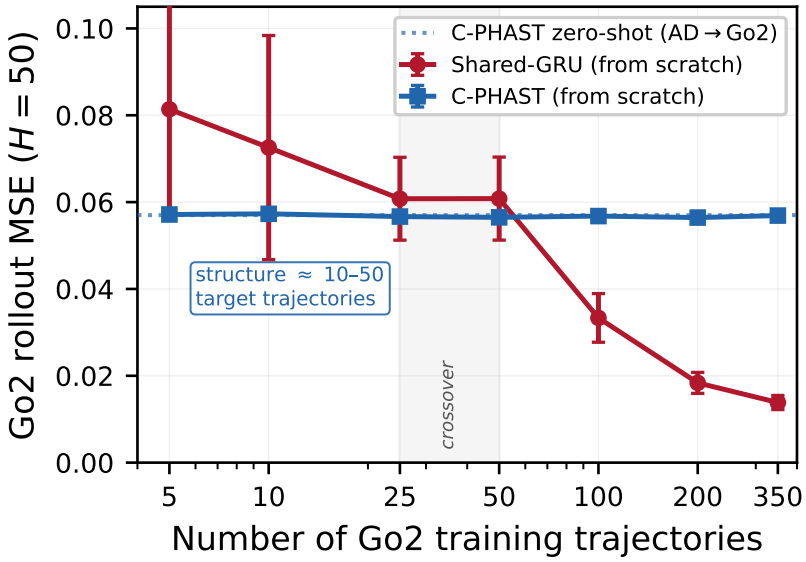

Figure 20. Data efficiency: Go2 rollout MSE (H=50) versus number of Go2 training trajectories (N). C-PHAST exhibits a flat structural ceiling regardless of N, while Shared-GRU improves steadily and surpasses C-PHAST around N=25–50. The physics prior is worth ∼10–50 target trajectories.

Key findings:

- **C-PHAST has a fixed structural ceiling**: Go2 H=50 rollout MSE stays at ≈0.057 regardless of N (from 5 to 350 trajectories), with near-zero variance across seeds ($\sigma < 0.001$).
- **Shared-GRU improves monotonically**: from 0.081 at N=5 to 0.014 at N=350.
- **Crossover at** N≈25–50: GRU matches C-PHAST's performance with ∼25 trajectories and surpasses it beyond ∼50.
- **C-PHAST from scratch ≈ C-PHAST zero-shot from ANYmal-D**: the structural prior provides equivalent performance whether instantiated from source-domain pretraining or from-scratch target training. In takeover rollout, most of C-PHAST's zero-shot advantage appears to come from its structural prior rather than source-domain parameter adaptation. The information ladder of Appendix I.5 refines this reading: with a starved burn-in the prior is all that acts, while with an informative burn-in and refreshed rollouts the trained model tracks the unseen robot's gait far below the persist level. A prior-only ablation (untrained/random-weight, analytical-damped, and constant-velocity baselines) run in the canonical regime is left to future work.

The practical implication: physics structure is worth 10–50 target trajectories. When target data is scarce ($N < 25$), C-PHAST dominates; when target data is abundant ($N > 100$), unconstrained models exploit their greater representational capacity.

I.5. Canonical Regeneration, Rollout Modes, and the Information Ladder

The Isaac artifacts behind Table 8 are regenerated end to end, because the original trajectory caches and checkpoints were not retained. The regeneration pins everything the result depends on: locomotion policies (1,500 PPO iterations, 4,096 environments, seed 42), collected trajectories (500 per robot, 200 steps), the model constructions, the matched training protocol, and both evaluation modes. All of it lives in one committed configuration, every reported number is written to a results manifest keyed by model, robot, seed, and rollout mode, and a single command reproduces the full table. The regenerated persist levels (0.045 on ANYmal-D, 0.152 on Go2) closely match the originally reported 0.043 and 0.131, and the qualitative pattern of the earlier table survives unchanged: the same transfer gaps for the pH rows, the same collapse of the graph baselines, and the same strong-source, weak-target signature for the recurrent ones.

The burn-in window is an information budget. The takeover context is not a nuisance parameter; it is the information the velocity observer works with. From a single frame, joint velocities are unobservable, so the observer supplies no momentum and the damped Hamiltonian flow settles toward stance. Rollout error then sits at the persist level, not because the learned dynamics are wrong but because the gait phase was never delivered. Five frames of burn-in recover the reported transfer skill on Go2 (rollout MSE 0.073 against a persist level of 0.152). A ten-frame window carries the gait open loop, with the model visibly walking the unseen robot before its amplitude decays (0.078 against a persist level of 0.136 in that window). Refreshed every five steps, the same weights track the gait (0.015), and refreshed every step they nearly interpolate it (0.001). Figure 21 traces this curve, and Figure 22 shows the corresponding trajectories on the six most transfer-sensitive joints. What changes across the rungs is the information delivered to the observer, not the learned flow.

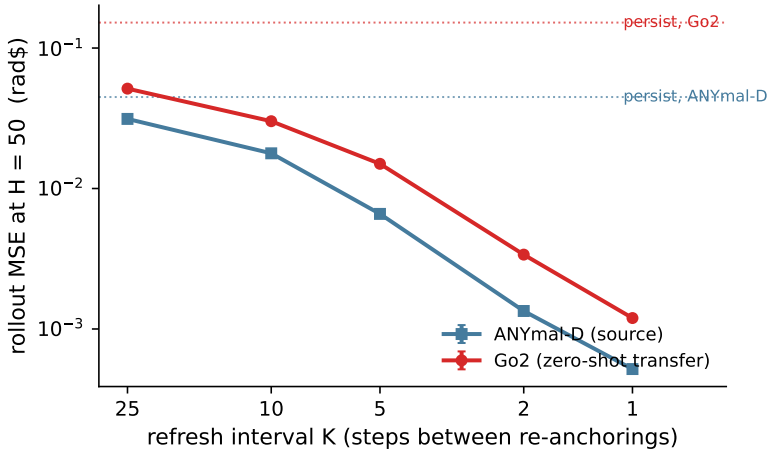

Figure 21. Rollout error at $H{=}50$ as a function of the refresh interval K (canonical checkpoints, 3 seeds, mean $\pm$ 1σ). The same trained C-PHAST moves from the persist level under starved takeover to near interpolation under per-step refresh; the deployment default $K{=}5$ sits well below both persist levels. Dotted lines: persist in the corresponding window.

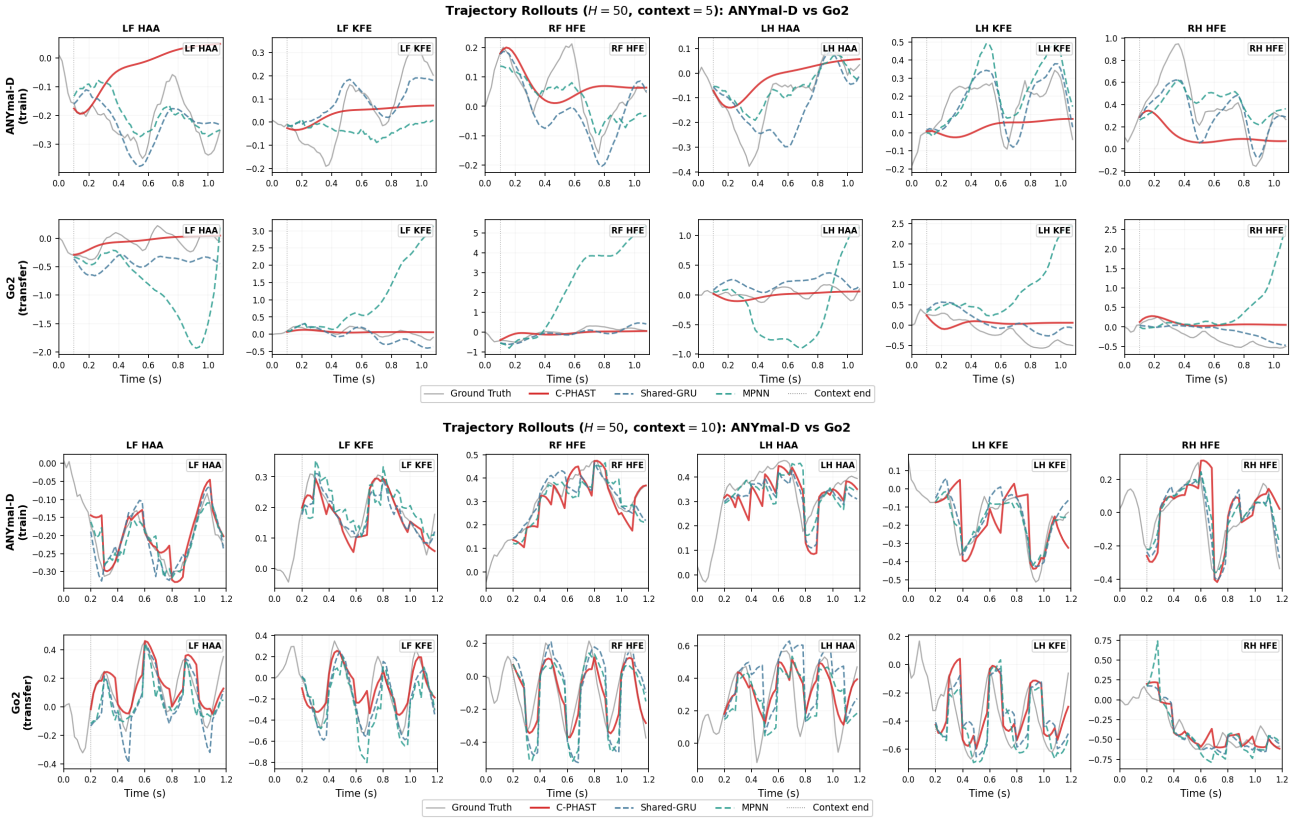

Figure 22. Ground truth, C-PHAST, Shared-GRU, and MPNN joint trajectories on both robots for the median-error test trajectory (canonical seed 42). Top: takeover rollout with the five-step burn-in. Bottom: the same models refreshed every five steps. In takeover, C-PHAST remains bounded while the baselines drift into plausible but wrong gaits; under refresh, C-PHAST tracks the zero-shot Go2 gait. Side-by-side robot renderings of these trajectories are provided as supplementary videos.

The structured recipe is mode sensitive. The full C-PHAST recipe (200 epochs, energy regularization, observer velocity supervision) was previously reported as a single row; the canonical regeneration shows its effect depends on both the burn-in and the rollout mode, which is why it is reported here rather than inside the matched table. With a one-frame context it equalizes the two robots (0.073 on ANYmal-D and 0.076 on Go2, a $1.0\times$ ratio), at the cost of source accuracy. With the standard five-frame burn-in it hurts transfer (0.149 on Go2 against 0.073 for the matched recipe), and under refresh it is also behind (0.020 against 0.015). Two observations explain the pattern. The energy and observer terms act as regularizers that damp the model's reliance on inferred momentum, which helps exactly when momentum is unobservable and hurts once it is supplied. And the observer term does not improve velocity estimation: supervising the observer toward ground-truth velocities increases its velocity error during training, because the dynamics have co-adapted to the observer's own smoothed estimate. We therefore read the structured recipe as gap-flattening regularization for starved-context regimes, and use the matched recipe everywhere else.

Scope. The few-shot adaptation, target-data scaling, jerk, and observer-ablation studies in this appendix were run in the original data regime, whose artifacts predate the regeneration; their qualitative conclusions are consistent with the canonical regime, and their absolute numbers should be read against the original persist levels quoted in each table.

J. Real-Robot and Decision-Interface Diagnostics

The previous appendix section explains the fixed-layout legged transfer result and its target-data budget. This section collects the robotics evidence whose purpose is different: it tests what the deployed predictor exposes to downstream decision layers. Spot logs validate real-robot telemetry-to-action-card forecasting; the Isaac push studies validate typed model-predictive warning and candidate ranking when alternative futures can be branched in simulation. The two subsections should therefore be read as complementary interface tests: Section J.1 fixes the real-robot shadow-mode information contract,

while Section J.2 tests the same factor interface when simulator branches are available.

J.1. Real Spot Shadow-Mode Action-Card Contract

The Spot-with-arm evidence uses logged ROS bags collected in a lab environment under human teleoperation. C-PHAST is evaluated as a shadow model: it observes telemetry and possible-next-behavior descriptors, predicts typed physical consequences, and writes objective-conditioned action cards beside the teleoperated run. The data are converted into overlapping windows with a two-second history and a five-second future horizon. Each row supervises the future that was actually executed in the log. Thus Spot serves as the real-robot telemetry-to-action-card forecast: the log supplies physical supervision for the executed future, while branch execution and matched-state alternative rollouts are tested by the Isaac experiments. Table 36 gives the scale of this logged dataset and the history/future windows used by the reported result.

Table 36. **Spot-with-arm window dataset.** The held-out split is by windows after filtering unusable or calibration-test segments.

Quantity	Value
usable teleoperated episodes	6
typed possible-next-behavior windows	1,070
training windows	897
held-out windows	173
history length	2 s
future-consequence horizon	5 s
reported query inputs	history + numeric behavior descriptors

Because Appendix J combines real logs and simulator branches, Table 37 separates the role and hierarchy of each artifact before the Spot details accumulate. The Spot row is real-robot evidence for telemetry-to-action-card prediction under logged replay; the Isaac transfer and push rows carry the branchable PHASTCore rollout evidence.

Table 37. **Benchmark roles and hierarchy for the shared query interface.** Each artifact stresses a different part of the C-PHAST claim: simulated branch execution, typed physical-critic scoring, or real robot telemetry-to-action-card forecasting.

Experiment	Query mode	Hierarchy used	Real robot	Validated role
Isaac transfer	branchable q-only rollout	four 3-DOF leg PHASTCore rollout blocks (HAA/HFE/KFE); floating base excluded	no	cross-robot rollout accuracy and pH/compositional transfer
Push critic	branchable q-only rollout plus critic	same Go2 four-leg rollout hierarchy, with support/contact and envelope factors read from candidate futures	no	typed branch scoring and early physical warning in simulation
Spot shadow mode	logged replay windows	interface blocks for base, leg support/contact, arm, camera/depth, and battery/power; target hierarchy for follow-up closed-loop hardware rollout	yes	real telemetry-to-action-card forecasting

The Spot hierarchy used here is an interface hierarchy. The declared blocks organize real telemetry into typed consequences and action-card factors. This is why Spot is paired with Isaac branch execution: the hardware logs validate the real sensing and decision interface, while the simulator experiments validate same-state alternative futures under PHASTCore rollout.

The main subtlety is the Spot input contract. C-PHAST is not allowed to solve the task by memorizing primitive names: the reported model receives recent telemetry plus numeric behavior descriptors, not a one-hot primitive label. Table 38 separates four quantities that are easy to conflate: history observed before the query, behavior descriptors that a planner could provide, replay-only descriptor fields measured from the logged future, and consequence targets that must be predicted.

Table 38. **Spot input-contract audit.** The window-split result tests the replay version of the planner-facing interface. A deployed confirm-to-execute planner would supply behavior-descriptor columns from intended primitive parameters rather than measured future telemetry.

Quantity class	Examples in the Spot artifact	Use in this paper
Decision-time history	recent speed/contact, hand motion, gripper state, image/depth quality, TF/depth novelty summaries	observed before the query; valid model input
Command-like behavior descriptors	commanded base speed and turn rate; gripper opening; primitive prototype descriptor	intended planner inputs; logged values are used as replay proxies
Future-derived behavior descriptors	realized yaw change, realized hand speed, realized hand turn over the five-second future window	used only in logged replay; must be supplied by a primitive library or forecasted before live use
Predicted consequence targets	work, support/contact, battery draw, collision/contact proxy, sensing quality, depth novelty, residual-like terms	unavailable before execution; predicted by the typed consequence model
Visualization / metadata	behavior labels, episode type, trim markers	used for analysis and plots; not part of the main reported model input

This contract gives two related query types. The held-out objective-score task is logged-window consequence prediction: each dataset row uses the executed teleoperation future as supervision. The same-history action-card query is a model query: it fixes one observed history and swaps in four prototype behavior descriptors, namely fast base loop, slow base loop, move-and-scan, and static arm scan, estimated from the training episodes. That query demonstrates objective switching from one state-history context; Isaac experiments provide matched-state branch execution when actual alternatives must be run.

C-PHAST separates raw observables, predicted typed consequences, planner factors, and objective scores. Raw robot measurements provide evidence; the consequence model predicts channels such as work, support/contact, sensing quality, collision/contact proxy, and residual-like terms; planner factors turn those named channels into decision quantities such as costs, rewards, barriers, or hard envelopes; objective weights then produce an action-card score. The same typed factor vector is reused for five objectives: efficient motion, safe motion, mapping, balanced operation, and inspection.

Typed factors versus raw observables. The action-card interface does not score raw telemetry directly. Each proposed behavior first produces typed readouts such as work, battery draw, support/contact quantities, collision/contact residuals, sensing quality, and depth/TF novelty. Table 39 records the concrete factor families used in the Spot artifact and the decision role each family plays.

Table 39. **Spot observable-to-factor examples.** The real-robot artifact instantiates the Figure 7 interface with concrete robot telemetry and behavior descriptors. Objective scores scale each planner-facing factor by its training-split 10–90 percentile range before applying objective weights.

Factor family	Construction from predicted typed consequences	Decision role
Progress	future base path length in the horizontal plane	reward for traversal
Work / energy	positive joint work $\sum_t \max(\tau_t^\top \dot{q}_t, 0)\Delta t$; arm and locomotion splits when available	physical cost
Battery draw	negative battery current under discharge, aggregated over the horizon	electrical energy cost
Support/contact	support-loss fraction, full-support fraction, contact-count deficit, foot/contact proxies	safety envelope and warning
Wrench/load	mean force and torque norms from available wrench or effort channels	contact/load cost
Collision/contact residual	hand or contact-envelope violation fraction; residual-like contact mismatch	veto or warning
Motion/aggression	speed, yaw motion, commanded speed and turn magnitude	blur risk, support risk, and progress estimate
Sensing quality	image frame/camera counts, sharpness, contrast, brightness, entropy, edge density	inspection quality
Depth/TF novelty	valid depth fraction, point count, observed/new voxel counts, novelty fraction, coverage rate	map gain

Planner factors then turn those readouts into decision quantities with declared semantics: source channel, units or normalization, horizon aggregation, target or limit, and role as a soft cost, reward, residual, barrier, or hard constraint. This separation is what makes objective switching transparent. An energy-efficient objective can put high weight on positive work and battery draw; a mapping objective can favor coverage or image/depth novelty while keeping support and collision factors as hard constraints; a protection objective can use residual and envelope factors as vetoes rather than as small terms in a weighted sum. Table 40 makes those five objective consumers explicit.

The five objectives used in the Spot result are fixed linear consumers of the planner-factor vector. Efficient motion rewards progress while penalizing work, battery draw, command aggression, support loss, and collision. Safe motion places stronger penalties on support loss, collision, force, and aggressive commands. Mapping rewards progress, yaw/hand motion, image quality, and depth novelty while penalizing speed, work, battery draw, support loss, and collision. Balanced operation trades progress, sensing motion, energy, support, force, and collision. Inspection emphasizes image and depth quality, stable viewpoint changes, hand motion, and hand turn, with penalties for speed, near-depth risk, support loss, collision, and

work. Changing objectives therefore changes the weights applied to the same typed factor vector rather than retraining the dynamics or consequence model.

Table 40. **Objective weights for Spot action cards.** After quantile scaling, each objective is a transparent weighting of typed planner factors. Positive entries reward a predicted consequence, while negative entries penalize cost or risk.

Objective	Rewards	Penalties / veto pressure	Intended action card
Efficient	progress $+1.0$	support loss -1.0, collision -1.0, command aggression -0.8, work -0.5, energy/progress -0.4, battery draw -0.3	move if progress is worth the physical cost
Safe	progress $+0.2$	support loss -2.0, collision -1.5, force -0.6, command aggression -0.5	stay inside the support/contact envelope
Mapping	progress $+0.7$; yaw/hand motion $+0.6/+0.8$; image/depth novelty and quality $+0.3$–$+0.6$	support loss -1.0, collision -0.8, speed -0.5, work/battery -0.3	prefer informative viewpoints without unsafe motion
Balanced	progress $+0.7$; yaw/hand motion $+0.3/+0.4$	support loss -1.2, collision -0.8, force -0.6, work -0.4, energy/progress -0.25	trade progress, sensing, and physical risk
Inspection	image/depth quality $+0.4$–$+0.9$; hand motion $+1.0$; yaw motion $+0.8$; progress $+0.25$	support loss -1.2, collision -1.0, speed -0.8, near-depth risk -0.2, work -0.15	slow local scan with high visual/depth fidelity

Table 10 gives the compact real-robot evidence ledger in the main text. The appendix expands that ledger in four steps: Table 41 reports the held-out window split, Table 42 tests leave-one-episode-out generalization, Table 43 checks which query inputs carry signal, and Table 44 compares typed readouts against direct scalar-objective training.

Table 41. **Held-out objective-score agreement on Spot logs.** All rows use recent history plus numeric behavior descriptors without primitive identity labels. Parentheses give paired-bootstrap 95% confidence intervals over the 173 held-out windows. Best value per metric is shown in **bold green**.

Objective	Pearson	Spearman	MAE
Efficient	0.744 [0.682, 0.803]	0.753 [0.681, 0.811]	0.213
Safe	0.907 [0.873, 0.935]	0.875 [0.833, 0.903]	**0.106**
Mapping	0.919 [0.894, 0.939]	0.900 [0.861, 0.926]	0.370
Balanced	0.948 [0.931, 0.962]	0.939 [0.901, 0.961]	0.140
Inspection	0.884 [0.846, 0.914]	0.836 [0.767, 0.886]	0.540

Table 42. **Leave-one-episode-out Spot stress test.** Training on five teleoperated episodes and holding out the sixth reduces agreement relative to the window split, as expected when entire behavior episodes are unseen. The positive correlations indicate that the typed query signal persists, but the window split remains the main interface-evidence result. Best per metric is shown in **bold green**.

Objective	Pearson	Spearman	MAE
Efficient	0.538	0.412	0.324
Safe	0.641	0.590	0.382
Mapping	**0.750**	**0.617**	0.318
Balanced	0.644	0.517	**0.317**
Inspection	0.686	0.590	0.473
Mean	0.652	0.545	0.363

Table 43. **Query-input ablation.** Both current history and behavior descriptors carry signal; primitive identity alone does not. Best per metric is shown in **bold green**.

Inputs	Mean Pearson	Mean Spearman	Mean MAE
history only	0.767	0.737	0.360
behavior descriptors only	0.780	0.778	0.334
history + behavior descriptors	**0.880**	**0.860**	**0.274**
primitive label only	-0.120	-0.054	0.740
history + descriptors + label	0.854	0.844	0.313

Table 44. **Typed readouts versus direct scalar objectives.** Typed consequences preserve scalar ranking quality while exposing the reason for the ranking. Best per metric is shown in **bold green**.

Training target	Mean Pearson	Mean Spearman	Mean MAE
typed consequences then score	**0.880**	0.860	**0.274**
direct scalar objective	0.871	**0.875**	0.330

For a fixed observed history, the ranked future changes when the objective changes. In the representative query summarized above, efficient and safe objectives choose a slow-loop candidate, while mapping, balanced, and inspection objectives choose a move-and-scan candidate. This is the operational reason to expose typed factors rather than a single learned score: a planner or operator can change the task objective while reusing the same consequence predictions.

J.2. Model-Predictive Physical-Critic Details

Section 4.2 reports the main physical-critic evidence: a directional simulator-executed sweep in which C-PHAST predicts unsafe futures before all failing Go2 push rollouts. This appendix gives the candidate-level interpretation of those runs and explains the information contract between the learned rollout model and the external candidate generator. The experiment isolates the online world-model interface: C-PHAST evaluates supplied futures, while the controller layer decides which future motions are available and what safety certificate, if any, wraps their execution. It tests whether the deployed C-PHAST rollout operator can be executed online as the critic in Eq. (27): at each timestep, a small finite set of candidate control sequences is rolled forward, scored, and either ranked or rejected as unsafe. Figure 23 records the corresponding critic-loop schematic. Table 45 then separates the possible consumers of the same typed readout vector: shadow monitoring, finite-candidate ranking, safety vetoes, lexicographic selection, and sampling-based MPC.

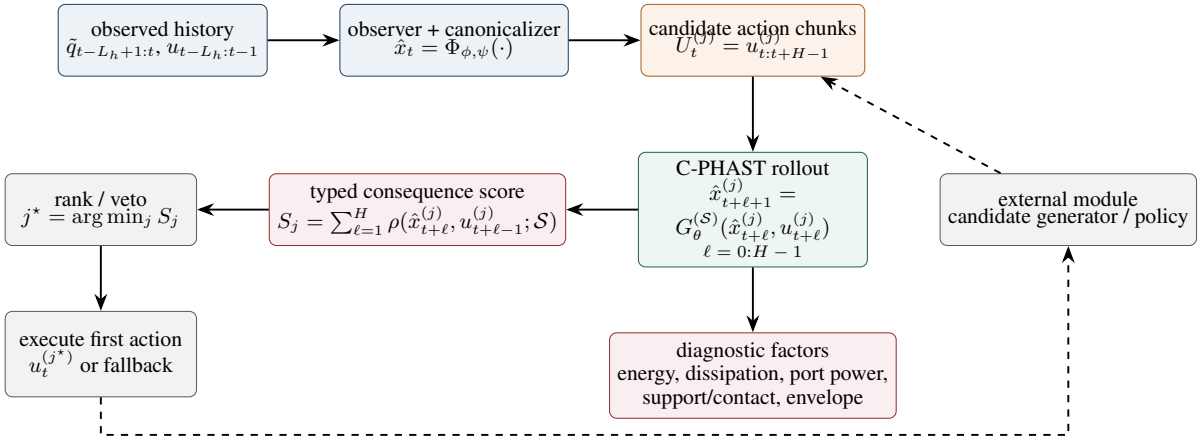

Figure 23. **C-PHAST as a model-predictive world-model critic.** The learned C-PHAST operator remains a rollout model, but downstream decision loops can query it on candidate action chunks. The critic ranks or vetoes candidate futures using typed physical consequence scores; the candidate generator or recovery policy is external to C-PHAST.

Table 45. **Planner modes supported by the same C-PHAST factor interface.** The learned rollout model produces the same typed readout vector in every mode; only the consumer and decision rule change.

Mode	Decision rule	Typical use
Shadow monitor	compute typed factors, log warnings, send no command	real-robot calibration and operator-facing diagnostics
Finite-candidate ranker	minimize a weighted soft score after normalization	choose among supplied action chunks or contact/support hypotheses
Safety veto	reject candidates whose barrier or envelope factors exceed thresholds	prevent unsafe continuations before optimizing task performance
Lexicographic selector	apply priorities such as safe first, then task progress, then energy	avoid hiding safety violations inside a scalar reward
Sampling/MPC consumer	sample or optimize candidate sequences, then query the same factors	MPPI/CEM/MPC layers and learned action generators

Setup. We use the Go2 Isaac Lab policy rollout with a lateral push applied at step 75. Failure is the simulator-level failure predicate used for the closed-loop filmstrips: fall, support loss, or slip depending on the configured run. The decision layer evaluates candidate control sequences over a short horizon and applies the selected sequence for 12 control steps before replanning. The simulator-executed score contains simple physical-envelope factors available in the current instantiation: predicted root-height margin, predicted maximum tilt, and predicted energy growth. The aggregate directional sweep in Table 9 separates *unsafe-future signals* from *action changes*: the former asks whether the policy continuation is predicted unsafe, while the latter asks whether the supplied candidate library includes a lower-risk alternative. Offline diagnostic artifacts additionally expose typed warning channels such as modal surprise, joint residuals, block residuals, and interconnect power; these support the typed-residual interpretation but are kept out of the main table for readability. Figure 24 is a lead-time plot: it asks when the warning or unsafe-future signal appears relative to simulator-level failure. Table 46 gives the corresponding event rows and keeps recovery separate from early warning.

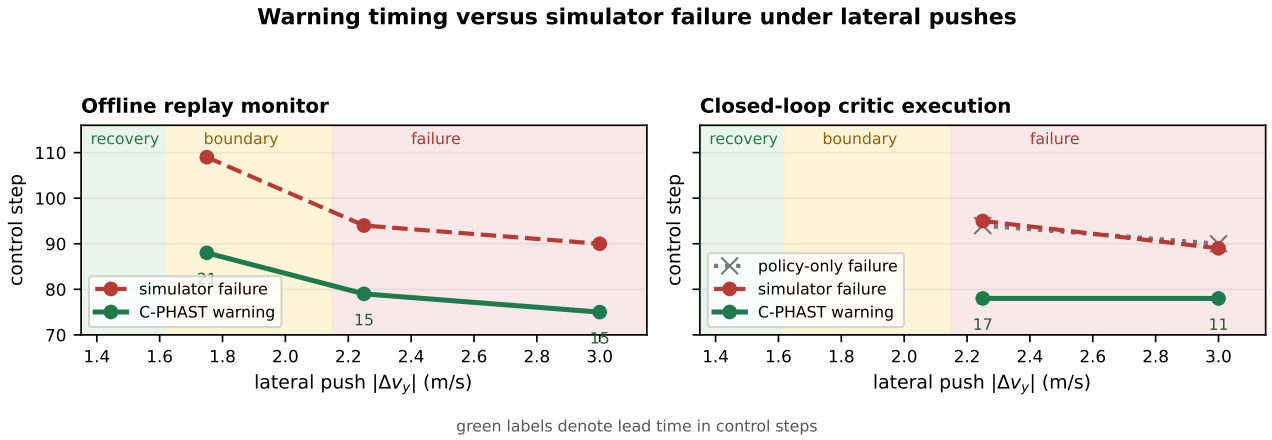

Figure 24. **Warning timing versus simulator-level failure under lateral pushes.** Left: offline replay asks whether C-PHAST consistency warnings appear before failure on logged policy rollouts. Right: closed-loop execution asks whether the model-predictive critic emits an unsafe-future signal before simulator-level failure. Green numbers denote lead time in control steps. The intended readout is temporal: warnings appear before failure; recovery depends on the supplied candidate library and is separated in Table 46.

Table 46. **Isaac Lab model-predictive physical-critic experiment.** Policy failure is the first simulator-level failure step in the policy-only rollout. First unsafe is the first step at which any candidate future violates the C-PHAST predicted safety envelope. First action change is the first step at which the decision layer selects a lower-risk non-nominal candidate. The first unsafe-future and action-change signals appear at step 78, shortly after the lateral push at step 75 and before simulator-level failure. Candidate control sequences are supplied externally; the result evaluates the critic interface that ranks and rejects those futures.

Run	Push	Fail	Unsafe	Action	Outcome
Policy only	1.50 m/s	—	—	—	no failure in recorded window
Policy only	2.25 m/s	94	—	—	fails after push
Policy only	3.00 m/s	90	—	—	stronger push fails earlier
Critic: hold-action candidate	2.25 m/s	94	78	78	selects hold-current-action sequence; fails at step 95
Critic: posture candidates	2.25 m/s	94	78	78	selects crouched-posture sequence; fails at step 94

First action-change diagnostic. Table 47 shows the first action-change event for the two C-PHAST critic variants. The policy continuation is already marked unsafe at step 78. The selected alternative has slightly lower score, but remains unsafe under the current envelope; at this state, the supplied candidate library lacks a recovery future. The useful observation is both temporal and semantic: the critic detects the impending failure 16–17 control steps before simulator-level failure and identifies that all supplied futures remain outside the safe envelope.

Table 47. **First C-PHAST action-change event at step 78.** Scores are lower-is-better model-predictive risk scores over the candidate future. Pred. min root-z rel. is the predicted root-height margin relative to the learned safe envelope; Pred. max tilt is the maximum predicted body-tilt proxy over the same horizon. The selected alternatives improve the score slightly, but do not move the predicted future back inside the safe envelope; hence the query layer correctly reports that the supplied alternatives are still unsafe. Lower selected scores are shown in **bold green**.

Candidate family	Selected sequence	Policy	Selected	min root-z	max tilt
hold-action	hold-current-action sequence	0.1111	**0.1064**	−0.1219	0.2197
posture	crouched-posture sequence	0.1111	**0.1103**	−0.1219	0.2212

Interpretation. The current candidate set contains policy continuation, holding the current action, scaled-policy actions, zero action, and simple posture sequences. Once the robot has accumulated large angular momentum, those candidates often occupy the same unsafe basin. C-PHAST therefore provides the telemetry-to-decision layer: it exposes the predicted envelope violation early and tells the downstream planner that the supplied alternatives are inadequate. A learned contact-port-conditioned action generator is the natural next component for expanding the candidate library under large pushes.

Support-choice ranking audit. The closed-loop sweep above tests early warning and online execution. The next question is different: if a planner supplies several recovery-like futures from the same pushed state, can C-PHAST help choose the one that survives longest? We answer this with an offline branch audit. For each pushed state, the simulator executes every supplied candidate and reveals the best surviving candidate after the fact. The ranker does not see that answer; it must choose using either action-label statistics, pre-state context, or C-PHAST's predicted typed factors.

From three branch-start times and 18 linear/angular root-velocity perturbations, we force 8 candidate labels per context, yielding 432 simulator branches and 34,176 C-PHAST candidate-factor rows. The candidate set contains mirrored support/catch primitives plus nominal alternatives, so the best choice should depend on push direction and robot state, not only on the candidate name. We report *survival regret*: the number of simulator steps lost by the selected candidate relative to the best candidate in the same branch group. A regret of zero means the selected candidate ties the best survivor. Table 48 compares four information levels: an action-label prior uses only the average survival of each candidate label, context uses push and pre-state features but no C-PHAST factors, prior+context combines those two, and C-PHAST factors use predicted support/contact, energy, port-work, damping-spectrum, and viability channels. In the second column, "factors" means the C-PHAST predicted future-factor summary, "candidate" means the action label is also included, and "viability" means terminal physical-margin channels such as root-height margin, tilt, energy growth, support-port magnitude, and damping-spectrum summaries.

Table 48. **Support-choice ranking audit.** Each row asks how much survival is lost when a ranker chooses among supplied Go2 support/catch candidates from the same pushed state. Lower survival regret is better; zero means the chosen candidate ties the best simulator-executed branch. C-PHAST future factors reduce regret beyond action-label averages and pre-state context on the all-candidate and paired-support settings. One support primitive remains dominant in 0.59 of all-candidate groups, so the audit is not solved by candidate identity alone; it evaluates whether physical evidence helps choose among supplied motions. Best regret per row is shown in **bold green**.

Candidate set	C-PHAST factors used	Action-label prior	Context only	Prior+context	C-PHAST factors
All support candidates	factors + context	12.31	13.35	12.31	**8.11**
Paired support modes	factors + context + candidate	5.70	5.57	5.69	**3.30**
Nominal vs support	factors + viability + context	1.76	6.54	1.76	**1.44**
Recovery candidates only	factors + context + candidate	5.70	5.57	5.69	**3.30**

The key comparison is C-PHAST factors against prior+context: if the push direction, pre-state, and candidate label already explained the choice, predicted physical factors would not improve regret. Instead, the factors reduce regret by 34.1% on all support candidates and 42.2% on paired support modes relative to the action-label prior. This result upgrades the critic evidence from early warning alone to physically informative ranking over finite candidate futures: the candidate generator supplies the alternatives, and C-PHAST supplies typed evidence for choosing among them.

J.2.1. Partial-Flag Damping-Spectrum Diagnostic

The physical critic above scores rollout futures using the full deployed predictor. This subsection asks a narrower diagnostic question: does the ordered damping spectrum exposed by the partial-flag PHASTCore parameterization provide a stable typed warning channel under push perturbations? The answer is yes, with an important qualification: the spectrum is a high-confidence channel for dissipative-modal regime change, not a complete failure detector.

Spectral coordinates. For a typed subsystem block b, let $q_t^{(b)} \in \mathbb{R}^{d_b}$ denote its local configuration at rollout step t, and let $D_b(q_t^{(b)}) \in \mathbb{R}^{d_b \times d_b}$ denote the learned momentum-damping operator from Eq. (115). Under the partial-flag specialization,

$$D_b(q_t^{(b)}) = d_{0,b} I_{d_b} + \sum_{r=1}^{r_b} \beta_{b,r}(q_t^{(b)}) k_{b,r}(q_t^{(b)}) k_{b,r}(q_t^{(b)})^\top, \qquad \beta_{b,1}(q_t^{(b)}) \geq \cdots \geq \beta_{b,r_b}(q_t^{(b)}) \geq 0, \tag{127}$$

where $d_{0,b} \geq 0$ is an isotropic damping floor, $r_b \leq d_b$ is the monitored damping rank, $\beta_{b,r}$ are nonnegative damping strengths, and the directions $k_{b,r}(q_t^{(b)}) \in \mathbb{R}^{d_b}$ are orthonormal for fixed (b,t). The ordering fixes the usual gauge freedom of a PSD factorization: without it, rotations inside a repeated eigenspace can change the factors without changing the physical damping operator. This is why the diagnostic is expressed in spectra and eigenspaces rather than in the raw factor coordinates. For example, if a damping operator can be written as $KK^\top$, then K and KQ represent the same physical operator for any orthogonal Q; comparing the columns of K directly would punish an arbitrary representative, and would also mix scale with direction. The physically meaningful objects are instead the operator D_b, its ordered strengths, and the subspaces spanned by its dissipative modes. In a robot push, motion in these objects can indicate that friction, contact loss, actuator damping, or support geometry has changed; in an electrical network, the analogous channel can point to conductance, line, or load mismatch. With it, each block exposes scalar spectral channels

$$\lambda_{b,r}(t) := d_{0,b} + \beta_{b,r}(q_t^{(b)}), \qquad g_{b,r}(t) := \lambda_{b,r}(t) - \lambda_{b,r+1}(t), \qquad \tau_b(t) := \operatorname{tr} D_b(q_t^{(b)}), \tag{128}$$

where $\lambda_{b,r}$ are ordered damping eigenvalues for $r = 1, \ldots, r_b$, $g_{b,r}$ are adjacent eigengaps for $r = 1, \ldots, r_b - 1$, and τ_b is the total damping trace.

Why gaps matter. Let $\bar{D}_b$ be a reference damping operator estimated from pre-push and recoverable windows, and write $\Delta_b(t) = D_b(q_t^{(b)}) - \bar{D}_b$. Let $\bar{U}_b$ be the reference eigenspace for the monitored damping modes and $\widehat{U}_b(t)$ the corresponding eigenspace of $D_b(q_t^{(b)})$. Let $\bar{g}_b$ denote the smallest eigengap separating this monitored eigenspace from its complement; in implementation, modes whose reference gap is below a small numerical tolerance are not used as ordered modal coordinates.

Then standard perturbation theory gives the qualitative rule (Davis & Kahan, 1970)

$$\|\sin\Theta(\widehat{U}_b(t),\bar{U}_b)\|_2 \;\lesssim\; \frac{\|\Delta_b(t)\|_2}{\bar{g}_b}, \tag{129}$$

where $\Theta(\widehat{U}_b(t),\bar{U}_b)$ is the diagonal matrix of principal angles between the current and reference subspaces. Thus operator drift is only interpretable as stable modal motion when it is measured relative to the eigengap. This is the reason for using gap-normalized spectral scores instead of only a raw matrix norm.

Threshold calibration. We calibrate scalar channels from normal-operation windows, here pre-push and recoverable-push windows, rather than hand-picking absolute thresholds. For a centered bounded scalar diagnostic channel Z_t accumulated over a window of length T, with variance proxy σ^2 and bound $|Z_t| \le M$, Bernstein's inequality gives

$$\Pr\left[\sum_{t=1}^{T} Z_t \ge a\right] \le \exp\left(-\frac{a^2}{2T\sigma^2 + 2Ma/3}\right). \tag{130}$$

For operator-valued deviations $X_t = D_b(q_t^{(b)}) - \mathbb{E}[D_b(q_t^{(b)})] \in \mathbb{R}^{d_b\times d_b}$, matrix Bernstein bounds the probability of large spectral-norm deviations (Tropp, 2012; 2026):

$$\Pr\left[\left\|\sum_{t=1}^{T} X_t\right\|_2 \ge a\right] \le 2d_b \exp\left(-\frac{a^2/2}{\nu_X + L_X a/3}\right), \tag{131}$$

where $\|X_t\|_2 \le L_X$ and ν_X bounds the matrix variance parameter. Equations (129)–(131) justify the diagnostic protocol at the level needed here: estimate a normal envelope for scalar spectral channels and gap-normalized operator drift, then warn only when the channel leaves that envelope by a chosen margin. Here the *normal envelope* is the empirical range of a diagnostic score on non-failing reference windows, specifically pre-push and recoverable-push windows. For each spectral channel, we compute the largest reference-window score and treat it as the normal operating ceiling for that channel. We use these inequalities as threshold-design guidance, not as an independent safety certificate, because rollout windows are correlated and the learned channels are estimated from finite data. In Table 49, the margins 1.10, 1.25, and 1.50 mean that the warning threshold is set to $1.10\times$, $1.25\times$, or $1.50\times$ that normal operating ceiling.

Table 49. **Partial-flag damping-spectrum diagnostic under Go2 pushes.** Results are summarized across three independently trained partial-flag C-PHAST seeds. Typed residual warnings and action-change events are measured on logged directional rollouts with 26 failing and 10 recoverable runs. Ordered damping-spectrum rows are measured on the simulator-executed directional sweep with 14 failing and 6 recoverable runs. Entries report mean count across seeds with the seed range in parentheses; false warnings are counted on recoverable runs. No row is highlighted because the spectrum thresholds trade warning coverage against conservatism.

Channel	Calibration	Failing runs warned	False warnings	Lead steps
Typed residual warning	learned thresholds	$23.3/26$ (23–24)	$0/10$	12.7 (11.8–13.3)
Action-change event	learned thresholds	$22.0/26$ (21–23)	$0/10$	5.2 (4.0–6.3)
Ordered damping spectrum	$1.10\times$ normal envelope	$12.7/14$ (12–13)	$0/6$	10.9 (8.2–12.2)
Ordered damping spectrum	$1.25\times$ normal envelope	$10.7/14$ (10–12)	$0/6$	9.7 (7.7–11.6)
Ordered damping spectrum	$1.50\times$ normal envelope	$8.7/14$ (8–9)	$0/6$	10.6 (8.0–12.2)

Interpretation. Across all three seeds, the ordered damping-spectrum channel produces zero false warnings on recoverable pushes. As the margin is tightened from $1.10\times$ to $1.50\times$, detection coverage falls from about $13/14$ failures to about $9/14$ failures, while false warnings remain zero. This is the expected operating curve for a conservative typed channel: stricter thresholds increase confidence at the cost of coverage. The channel is therefore useful for interpreting *which physical sector* has left its normal envelope. It is not a replacement for the full model-predictive critic, because some failures are dominated by support/contact or observation-side mismatch rather than by a dissipative-modal change.

K. Deployment-Difficulty Ladder: Exact Cell Assignments

Figure 3 summarizes deployment difficulty as a single-axis landscape. The concrete redeployment triggers behind each column, and the method competitive there, are given in the main text (Table 3); this appendix gives the exact, auditable content

Table 50. **Specification fingerprint.** For each deployment-stress column, whether each information component is supplied and fixed (●), changed at deployment but still supplied (○), or withheld and learned (×). This is the exact per-cell content shaded in the fingerprint panel of Figure 3.

Component	full spec	local law learned	layout/ count	topology changes	q-only sensing	decomp. unknown
local law	●	○	○	○	○	×
type library	●	●	●	●	●	×
target layout	●	●	○	○	○	×
target graph	●	●	●	○	○	×
port map	●	●	●	●	×	×
observation map	●	●	●	●	○	○
metadata	●	●	●	●	●	●
target data	●	●	●	●	●	●

behind the figure's columns: the per-cell specification fingerprint (Table 50). The ordering axis is the deployment *information budget* $\mathcal{I} = (\text{local law},\ \text{type library},\ \text{target layout},\ \text{target graph},\ \text{port map},\ \text{observation map},\ \text{metadata},\ \text{target data})$; each column withholds or changes more of $\mathcal{I}$ than the one before it.