R-APS: Compositional Reasoning and In-Context Meta-Learning for Constrained Design via Reflective Adversarial Pareto Search

João Pedro Gandarela[1,2] **Thiago Rios**[3] **Stefan Menzel**[3] **André Freitas**[1,4,5]
[1]Idiap Research Institute, Switzerland [2]École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
[3]Honda Research Institute Europe, Germany
[4]Department of Computer Science, University of Manchester, UK
[5]National Biomarker Centre, CRUK-MI, University of Manchester, UK
firstname.lastname@idiap.ch firstname.lastname@honda-ri.de

Abstract

Large language models (LLMs) are fluent reasoners on open-ended tasks, yet on *constrained design* problems, and more broadly in *agentic* settings where an autonomous system must plan, use tools, and act over extended horizons, fluency does not translate into reliable delivery. We trace this gap to three coupled structural failures that affect agentic AI models in constrained domains: failures propagate without localization, worst-case perturbations are not evaluated, and accumulated knowledge is never invalidated. We argue that these failures share a single root cause, namely that abductive, counterfactual, meta-inductive, corrective, and inductive reasoning pull a shared context in incompatible cognitive directions. We introduce **Reflective Adversarial Pareto Search (R-APS)**, an agentic AI method and, to our knowledge, the first method that addresses all three failures jointly through *reasoning-mode decomposition*: a novel design principle that allocates each of the five reasoning modes its own cognitive context and orchestrates their interaction across three timescales, *intra-stage* staged compositional reasoning with *typed validation critic* for failure localization, *intra-episode* sensitivity-guided counterfactual stress-testing of robustness as a first-class Pareto objective, and *inter-episode* meta-inductive rule extraction with explicit invalidation constituting persistent agentic memory. Crucially, R-APS achieves these gains with a *frozen* reasoning LLM, without parameter updates or fine-tuning, and purely through structured protocol design. We instantiate and evaluate R-APS on *planar mechanism synthesis*, a fundamental engineering problem underlying robotics, prosthetics, and mechanical design, in which every candidate output is checked by a kinematic solver. On 32 target trajectories (6 standard curves and the 26 English-alphabet letters), R-APS produces robustness certificates $3.5\times$ tighter than uniform-perturbation baselines, accelerates iterations-to-first-archive-admission by 46% across episodes on the same shape, and attains a $2.1\times$ mean Chamfer-distance reduction over classical Enum+GA on standard curves while jointly controlling bar-count adherence and worst-case robustness. In a cross-backbone study, we further find that small reasoning-specialized 4B models are competitive with general-purpose 70B backbones inside the protocol, suggesting that well-structured agentic protocols can partially offset raw model scale. More broadly, the protocol is parameterized by domain interfaces (typed verification cascade, sensitivity primitive, refinement log) rather than mechanism-design knowledge, and the agentic loop generalizes naturally to SQL synthesis, circuit design, and robot motion planning.

Preprint.

1 Introduction

Consider an agent tasked with designing a planar mechanical linkage whose endpoint (Fig.1 point (P)) traces an ellipse. The agent proposes a bar topology, optimizes link lengths numerically, and checks the resulting trajectory against the target. The check fails, the trajectory overshoots the apex. So the agent discards *everything* and starts over: topology, parameters, the carefully reasoned decisions that got it this far. Yet the topology was valid. Only the optimizer needed adjusting. There was simply no mechanism to say so.

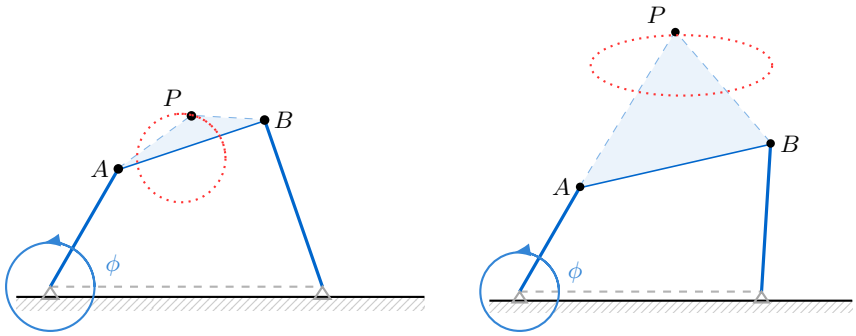


Figure 1: Six-bar linkage configuration before and after adjustment. The left panel shows the initial setup, and the right panel shows the adjusted configuration. The input crank rotates with angle ϕ, driven by the motor, while the red dashed curve near point P indicates the end-effector **trajectory**.

This scenario recurs across *constrained design* tasks more broadly: settings where an output decomposes into interdependent typed stages and every constraint must hold simultaneously, including database query synthesis Pourreza and Rafiei [2024], robot task planning Ahn et al. [2022], circuit design Thakur et al. [2023], and planar mechanism synthesis Song et al. [2025]. LLMs have become fluent reasoners on open-ended tasks Wei et al. [2022], Yao et al. [2023], and a new generation of *agentic AI* methods Xi et al. [2025], Wang et al. [2024] now wraps them in autonomous loops that plan, invoke tools, and adapt to feedback. Yet a persistent gap remains between how fluently a model can *discuss* a solution and how reliably an agentic method can *deliver* one: a reasoning failure here is not a stylistic blemish but an infeasible artifact, discarded in full.

Three structural failure modes compound when monolithic LLMs are used as agentic methods in this regime: (i) *failure propagation without localization*: a single violated constraint forces complete regeneration; (ii) *absence of robustness certification*: nominal performance says nothing about worst-case behavior under realistic perturbation, a critical reliability gap for any autonomously deployed agent; (iii) *monotonic heuristic accumulation*: methods that learn from experience never invalidate stale rules and discard refinement trajectories, causing knowledge quality to degrade over episodes. We argue that (i)-(iii) are coupled symptoms of current agentic LLM designs lacking a *reasoning protocol* that keeps abductive, counterfactual, meta-inductive, corrective, and inductive reasoning in cognitively distinct contexts rather than entangled in a single prompt. The central hypothesis is that failures (i)-(iii) can be addressed jointly by *reasoning-mode decomposition*, an explicit agentic protocol that allocates each mode its own specialized context:

> *Can a frozen reasoning LLM be turned into a reliable constrained-design agent purely by structuring its reasoning protocol, localizing its own failures, certifying its own robustness, and distilling its own refinement trajectories into transferable knowledge, without any parameter updates?*

Existing approaches give partial remedies but, to our knowledge, no prior method delivers the full bundle: Reflexion Shinn et al. [2024] and ReAct Yao et al. [2023] reflect verbally but treat each generation as atomic, so selective correction is impossible in principle and no robustness guarantee is produced; Voyager Wang et al. [2023] and ExpeL Zhao et al. [2024] accumulate skills monotonically and discard failure trajectories; general-purpose agentic frameworks such as AutoGen Wu et al. [2023] and tool-augmented agents Schick et al. [2023] decompose by workflow role rather than by reasoning mode, leaving the entanglement problem unaddressed.

We present **R-APS (Reflective Adversarial Pareto Search)**, an **agentic AI method** and the first, to our knowledge, to instantiate *reasoning-mode decomposition* as a complete protocol addressing all three structural failures jointly, organized along three interacting timescales (full mode taxonomy in Section 2.2). **(1) Intra-stage:** staged compositional reasoning (in the sense of typed decomposition, not Lake-style compositional generalization Lake et al. [2017]) with *typed validation critics* that attribute every failure to a specific named stage and route selective refinement only there, reducing constraint-adherence deviation from 1.9 to 1.5. **(2) Intra-episode:** sensitivity-guided counterfactual stress-testing in which a Designer (abductive), Adversary (counterfactual), and Critic (meta-inductive) play a min-max game with Sobol-screened directed perturbation, yielding robustness certificates $3.5\times$ tighter than uniform-perturbation baselines (0.132 vs. 0.458). **(3) Inter-episode:** meta-inductive rule extraction with explicit invalidation mines complete refinement trajectories including failures, serving as the agentic model's *long-term memory*; the resulting policy improves across episodes *without parameter updates*, delivering a 46% acceleration in iterations-to-first-archive-admission on repeated shapes.

Contributions.

1. A novel **reasoning-mode decomposition** principle for agentic AI, identifying five reasoning modes (abductive, counterfactual, meta-inductive, corrective, inductive) whose entanglement in a shared context is, to our knowledge, the first unified explanation of three otherwise-disconnected failure modes.
2. The first multi agent protocol, to our knowledge, to compose typed validation critics (intra-stage), counterfactual stress-testing as a Pareto objective (intra-episode), and meta-inductive rule extraction with *explicit invalidation* serving as agentic long-term memory (inter-episode) into a single closed agentic loop, all operating on a frozen LLM without parameter updates or fine-tuning.
3. Empirical validation on 32 shapes: a 46% inter-episode acceleration, a $2.1\times$ mean Chamfer-distance reduction over Enum+GA on standard curves, robustness certificates $3.5\times$ tighter than uniform-perturbation baselines, and 4B reasoning-specialized models competitive with 70B general-purpose models inside the protocol.
4. Architectural and empirical support for the decomposition: a falsifiable stage-by-mode responsibility map whose predicted ablation signature, each timescale owns one and only one of the three failure axes, is confirmed empirically.

2 R-APS: A Novel Reasoning-Mode Decomposition Method

R-APS is an **agentic AI method** grounded in a single hypothesis: *distinct reasoning modes optimize in incompatible cognitive directions, so reliable constrained design requires an agentic method that decomposes the reasoning modes into specialized contexts and orchestrates their interaction across an autonomous multi-step loop.* The method decomposes reasoning into five modes (abductive, counterfactual, meta-inductive, corrective, inductive) interacting along three agentic timescales: *intra-stage* staged compositional reasoning with typed validation critics (§2.3), *intra-episode* sensitivity-guided counterfactual stress-testing (§2.4), and *inter-episode* meta-inductive rule extraction with explicit invalidation, constituting the agentic model's long-term memory (§2.5). R-APS instantiates these modes through five specialized LLM agents forming a closed agentic loop: Designer (π_D, abductive), Critic (π_C, meta-inductive), Post-Opt Critic (π_{PC}, evaluative), Refinement (π_R, corrective), Meta-Analyst (MA, inductive); see Fig. 2 for how they interact and App. B for full agent profiles.

2.1 Problem Formulation

A planar mechanism is $M = \text{Assemble}(\tau, \theta, \mathcal{C})$, with topology $\tau \in \mathcal{T}$, parameters $\theta \in \Theta(\tau)$, and kinematic constraints $\mathcal{C}$. Given target $\mathcal{T}_{\text{target}}$, we evaluate mechanisms on two objectives: trajectory accuracy $f_1(M) = \text{CD}(\text{ICP}(\mathcal{T}(M)), \mathcal{T}_{\text{target}})$ via Chamfer distance (CD) with iterative closest point (ICP) alignment Besl and McKay [1992], and robustness $f_2(M) = \rho(M)$ measuring worst-case degradation under perturbations (§2.4). The goal is to construct a Pareto archive A of non-dominated mechanisms satisfying all kinematic constraints.

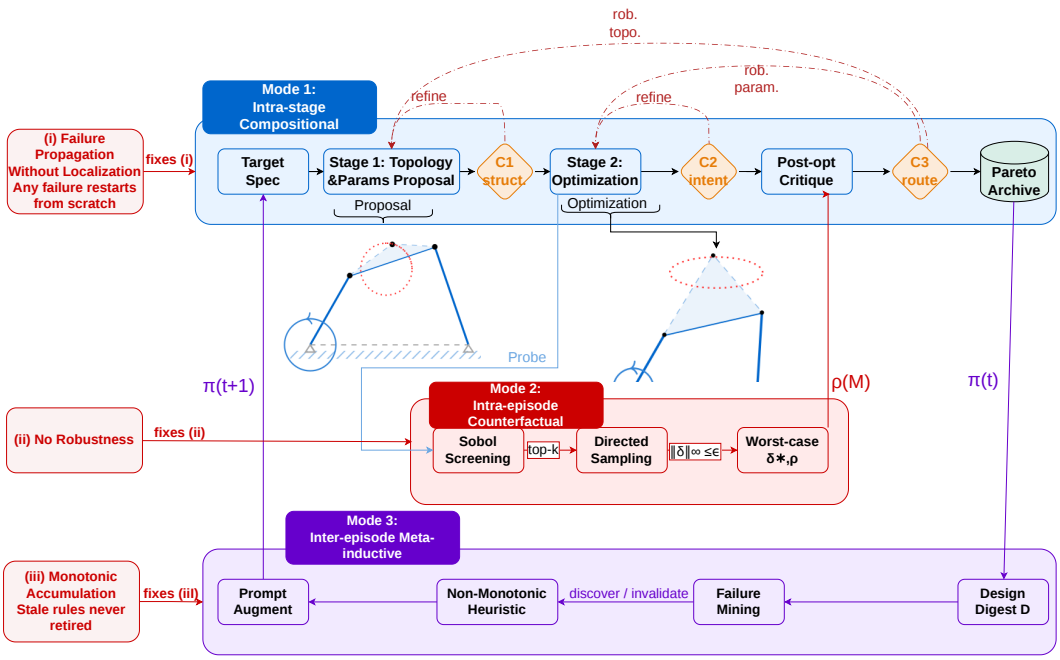


Figure 2: **R-APS: three-timescale reasoning-mode separation. Left (red):** the three structural failures each timescale addresses. **Mode 1 (Intra-stage)** decomposes design into typed stages with **typed validation critics** C1/C2/C3; selective refinement corrects only the diagnosed stage (i). The inset linkages show the intra-stage progression: left, the raw topology proposal (unconstrained geometry, no target); right, the same mechanism after optimization, with the coupler trajectory (red dotted ellipse) aligned to the target. **Mode 2 (Intra-episode)** runs Sobol-directed stress-testing, supplying the robustness certificate $\rho(M)$ that eliminates failure (ii). **Mode 3 (Inter-episode)** extracts and *invalidates* heuristics from refinement trajectories; the dotted purple loop delivers $\pi^{(t+1)}$ without parameter updates, eliminating failure (iii).

2.2 Five Reasoning Modes and Why We Decompose Them

Three modes descend from Peirce's classical triad [Peirce, 1878]: **abductive** (Designer, hypothesis generation), **deductive** (absorbed into the non-LLM structural validator/optimizer), and **inductive** (Meta-Analyst, rule distillation across trajectories). Two further modes are required by design loops which the classical triad does not address: **counterfactual** [Byrne, 2005, Pearl, 2009], the inference-under-disprovability step that grounds worst-case stress testing, and **corrective** [Lipton, 2004], the inference-to-best-explanation step specialized to localized-diagnosis repair. Each of the five addresses exactly one of the coupled failure modes (Section 1): corrective owns failure-propagation-without-localization, counterfactual owns absence-of-robustness-certification, and meta-inductive together with inductive own monotonic-accumulation-without-invalidation. The full per-mode summary table and the stage-by-mode responsibility map are in App. B (Tables 6, 7, and 8).

2.3 Compositional Design Method

R-APS instantiates the intra-stage timescale as a staged method separated by validation critics with selective refinement loops; full pseudocode is Algorithm 1 in App. A.

Typed validation critics. The critics make staged compositional reasoning actionable: each critic performs a *typed* diagnosis, attributing failure to a specific named stage rather than merely checking constraints. Critic 1 (structural) verifies hard kinematic constraints: link count and assemblability. Critic 2 (intent) checks whether the optimized trajectory semantically matches the specification via quantitative error thresholds and qualitative motion-primitive frequency matching (details in App. C). Failure diagnosis distinguishes topology-level errors (fundamental mismatch, $f_1 > 10\epsilon_{\text{traj}}$) from

optimization failures ($f_1 > \epsilon_{\text{traj}}$), eliminating the *failure propagation without localization* that plagues monolithic approaches.

Selective refinement. When a *typed validation critic* diagnoses failure at stage s, only stage s is corrected while all decisions from stages $1, \ldots, s-1$ are preserved, the structural guarantee that makes staged compositional reasoning more than decomposition. A topology failure triggers the next proposal from the pool; an optimization failure switches strategy (Broyden–Fletcher–Goldfarb–Shanno (BFGS) Broyden [1970], Fletcher [1970], Goldfarb [1970], Shanno [1970]↔particle swarm optimization (PSO) Kennedy and Eberhart [1995]↔Grid) and re-runs Stage 3 with the validated topology and parameters intact. A robustness failure (critic C3 in Fig. 2) is sub-typed by the failure fingerprint emitted by the adversary: ROBUSTNESS/TOPOLOGY (the nominal trajectory is itself off, despite worst-case behaviour being explored) is treated as a topology failure and re-proposed; ROBUSTNESS/PARAM (the design degrades sharply under perturbation) re-runs Stage 3 with the validated topology preserved and the fragile-parameter set surfaced to the optimizer (App. Table 9). Refinement depth is bounded at $D{=}20$ to prevent infinite loops.

2.4 Counterfactual Reasoning: Sensitivity-Guided Robustness Testing

Nominal trajectory accuracy is insufficient for practical deployment: real mechanisms experience manufacturing variations and assembly errors. R-APS realizes the *intra-episode counterfactual reasoning mode* through a two-phase protocol (full pseudocode in Algorithm 2, App. D; phase illustration in App. Fig. 5) in which the cognitively specialized constructive Designer and destructive Adversary engage in a min-max game. The two agents are kept in separate prompts precisely because abductive generation and counterfactual stress-testing are the canonical interfering pair (Section 2.2). Phase 1 uses Latin hypercube sampling (LHS) McKay et al. [1979] for even coverage of the tolerance hypercube and computes Sobol first-order indices Sobol [2001] to identify the most sensitive parameter dimensions; Phase 2 concentrates the remaining sampling budget on those dimensions around the high-loss centroid, locating the worst-case perturbation $\delta^* = \arg\max_{\|\delta\|_\infty \leq \epsilon} \text{CD}(\mathcal{T}(M_{\theta^*+\delta}), \mathcal{T}_{\text{target}}) - R_C(\theta^*+\delta)$. The robustness score $\rho(M) = \|\mathbf{f}_{\text{worst}} - \mathbf{f}_{\text{nom}}\| / \max(\|\mathbf{f}_{\text{nom}}\|, \varepsilon)$ enters the Pareto archive as a first-class objective alongside trajectory accuracy.

2.5 Meta-Inductive Reasoning: Inter-Episode Heuristic Lifecycle

The *inter-episode meta-inductive reasoning mode* is what distinguishes R-APS from methods that merely reflect within a single episode, and is the component that gives R-APS its character as a *learning agentic method* rather than a stateless pipeline. This mode constitutes the agentic model's **long-term memory**: it extracts reusable design knowledge from complete refinement trajectories, the full sequence of attempts, *typed validation critics* diagnoses, corrections, and outcomes, not just from successful designs, and makes that knowledge available to all agents in subsequent episodes without parameter updates. At each iteration t, the effective agentic policy is:

$$\pi^{(t)}(\cdot) = \text{LLM}\big(\cdot \mid \mathcal{P}_{\text{base}}, \mathcal{H}^{(t)}, \mathcal{D}^{(t)}, \mathcal{E}^{(t)}\big) \tag{1}$$

where $\mathcal{H}^{(t)}$ are learned heuristics, $\mathcal{D}^{(t)}$ is a compressed design digest (representative archive exemplars selected via uniform manifold approximation and projection (UMAP) embedding and hierarchical density-based spatial clustering of applications with noise (HDBSCAN) clustering), and $\mathcal{E}^{(t)}$ is an exclude list of repeatedly failed topologies.

The meta-inductive mode manages a **heuristic lifecycle**: new rules are synthesized from observed patterns, existing rules are updated with new evidence, and rules whose support erodes are *explicitly invalidated*, preventing the monotonic accumulation that degrades Voyager/ExpeL-style methods. Each heuristic carries a confidence score for conflict resolution. The meta-inductive mode operates over a different timescale than the intra-episode modes: its outputs feed the Designer and Critic prompts for subsequent episodes, so that inter-episode knowledge transfer happens in context rather than through weight updates, the key mechanism by which R-APS's agentic loop improves over time. Conceptually, this is an in-context analogue *in spirit* of inner-/outer-loop meta-learning Finn et al. [2017] (outer-loop rule evolution, inner-loop reflective correction), though the mechanism, context accumulation with explicit invalidation, is qualitatively different from gradient-based few-shot adaptation. Full details of design-space compression, failure-mode clustering, and resolution-pattern mining are in App. E.

3 Experiments

We evaluate R-APS on 32 shapes (6 standard curves plus 26 English letters) around five questions tied to the three structural failure modes and external competitiveness: **Q0** ablation non-contamination, **Q1** intra-stage, **Q2** intra-episode, **Q3** inter-episode, **Q4** vs. classical search (model-scale study in App. G.3).

3.1 Setup

Domain and metrics. We synthesize planar linkages whose end-effector trajectory tracks a target curve, on 32 shapes (6 standard curves: Circle, Ellipse, Line, LB, NACA airfoil, Parabola, plus the 26 letters of the English alphabet). We report two Pareto objectives jointly (Chamfer distance CD; worst-case sensitivity ρ_{sens}) plus the structural-adherence diagnostic ΔBars and a Norm. Dist. Index (per-run-median Chamfer $\times$ 100 over the median modular-baseline Chamfer; App. G.10).

Baselines and what each tests. We compare against three baselines, each selected to test a specific rival explanation rather than to provide general coverage. **Enum+GA** Cabrera et al. [2002] enumerates feasible 4- and 6-bar topologies and applies genetic algorithm optimization; it tests whether topology enumeration, the strongest non-LLM alternative, already solves the localization and robustness problems, making reasoning-mode decomposition unnecessary. **Modular LLM** Gandarela et al. [2026] is the closest faithful port of ReAct/Reflexion/Voyager/ExpeL to mechanism synthesis (feature-surface mapping in App. Table 10); it tests the rival hypothesis that workflow-role decomposition, assigning different LLM calls to planning, execution, and reflection, is sufficient and reasoning-mode decomposition adds nothing beyond that. **R-APS ablations** (*NoAdvMeta*: no adversarial testing, no meta-learning; *NoSelRef*: no selective refinement) test the specific mechanism claim: that each component is necessary for its assigned failure axis and does not substitute for the others. LLM are Llama-3.3-70B (general-purpose), Qwen3-4B (compact, reasoning-specialized), and Qwen3-30B-A3B (mixture-of-experts (MoE)), chosen to separate protocol-structure effects from model-scale effects.

3.2 Q0: Ablation Effects Do Not Cross-Contaminate

Removing each component degrades exactly one failure axis and leaves the other within noise (Table 4). Full R-APS attains the tightest adherence (1.5 ΔBars) and the lowest robustness score (0.132 ρ_{sens}). Removing the adversary+meta pair worsens robustness 3.5$\times$ (0.132 $\rightarrow$ 0.458) but leaves adherence within noise of Full; removing selective refinement worsens adherence 28% (1.5 $\rightarrow$ 1.9) but leaves robustness within noise. The non-overlap is the empirical signature predicted by the prescriptive mode-failure pairing (Appendix Table 7) and inconsistent with a single shared mechanism.

Modular LLM collapses three components and exhibits three failure modes simultaneously: on the modular-comparable shapes it reaches Norm. Dist. Index 68.3 and 2.0 ΔBars with no robustness certificate (Tables 3; baseline-feature surface in Appendix Table 10). All three R-APS variants beat Modular on the distance index, but adherence does not follow the same ranking: NoAdvMeta (2.2) does not beat Modular's 2.0 because the adversary+meta pair supplies the bar-count-constraining heuristics (the multi-objective signature examined in §3.4).

3.3 Q1: Most failures are absorbed at cheap early critics, and selective refinement rescues a meaningful fraction before expensive robustness checks are needed.

Typed critics concentrate failures at the cheapest stage; selective refinement converts most failed iterations into successful archive admissions (Tables 2, 3). Specifically, 59.2% of failures are absorbed at the topology critic, with only 2.4% reaching the expensive counterfactual screen; selective refinement then converts 13.0% of those failed iterations into archive admissions, yielding the tightest structural adherence (1.4 ΔBars) of any tested method while uniquely supplying a robustness certificate. Per-stage refinement budgets diverge sharply by critic identity (Table 2; App. Table 9), confirming *routed* rather than uniform retry.

Full R-APS achieves the best distance and adherence simultaneously, while uniquely supplying a robustness evaluation (Table 3). It attains ΔBars=1.4 (best) and Norm. Dist. Index=41.2 (best)

Episode bucket	Iterations-to-first-success
Episode 1	4.9 ± 0.5
Episodes 2–3	3.9 ± 0.3
Episodes 4+	2.7 ± 0.1

Table 1: Meta-inductive acceleration across episodes on the same shape (baseline preset). Acceleration Ep1→Ep4+: 46%, without parameter updates.

Failed Stage	**Share**	**Refines/fail**
Topology	59.2%	0.00
Param.	31.2%	1.30
Optimization	7.2%	1.96
Robustness	2.4%	0.66
Total	100.0%	0.56

Table 2: Failure localization by stage across the R-APS corpus (32 shapes). The cascade absorbs 59.2% of failures at the cheapest topology critic; only 2.4% reach the expensive adversarial screen. Refines/fail rising from the parameter stage confirms selective refinement reuses valid upstream decisions. Per-shape detail in Appendix Table 25.

Configuration	**ΔBars ↓**	**Norm. Dist. Index ↓**	**Robustness ↓**
Modular (Overall)	2.0 ± 0.0	68.3 ± 6.6	—
R-APS (w/o Adv+Meta)[†]	2.4 ± 0.2	61.0 ± 7.1	0.458 ± 0.078
R-APS (w/o Sel. Ref.)	1.6 ± 0.3	60.3 ± 47.5	0.116 ± 0.021
R-APS (Full)	1.4 ± 0.2	41.2 ± 11.3	0.132 ± 0.019

Table 3: Structural adherence and trajectory accuracy across modular LLM baselines and R-APS variants (all ↓ better). Parabola is excluded here because its raw distance is $\sim$100$\times$ larger than other shapes and would dominate row-level aggregates; it is broken out in Table 14. Robustness is computed over all available R-APS shapes; modular baselines lack robustness testing. Norm. Dist. Index uses the same canonical modular-baseline median as Tables 4 and 11, so values are directly comparable across tables. [†]R-APS (w/o Adv+Meta) ran on the alphabet-plus-parabola subset; the index base is unchanged, so the row is level-comparable despite the different shape mix. Best per column is **highlighted**.

at once. Within R-APS variants on the all-shapes pool, NoAdvMeta gives the worst adherence despite a lower raw distance: without robustness as an active Pareto axis, the optimizer compresses distance further but drifts on bar count (the multi-objective trade-off made explicit in §3.4). The cascade's failure mass aligns with the decomposition map (90.4% abductive, 7.2% corrective, 2.4% counterfactual; App. G.8).

3.4 Q2: The adversary+meta pair is what gives R-APS robust worst-case behavior.

Robustness is owned by the intra-episode mode; structural adherence by the intra-stage one, the two axes do not interfere (Table 4, Fig. 3). Removing the counterfactual+meta-inductive pair inflates ρ_{sens} by $3.5\times$, the largest single effect in the ablation, while removing selective refinement leaves robustness within noise. The adversary discovers worst-case failure modes that nominal evaluation hides, with nominal-vs-worst gaps of 0.038 median to 5.114 max on the top-5 most fragile designs (all ablation-induced shifts significant at $p < 10^{-4}$, Mann–Whitney U; App. G.14, G.17).

3.5 Q3: R-APS explicitly invalidates stale rules, allowing it to improve across episodes without parameter updates.

R-APS is the only system in our comparison surface that explicitly invalidates prior rules with cited counterexamples (App. Table 10; representative invalidations in App. Table 15). Of the 4 meaningful invalidations, 2 are cross-shape, each citing the specific design and shape that refutes the rule, distinguishing R-APS from Voyager/ExpeL-style libraries that accumulate monotonically and discard the corrective signal of failed trajectories.

On repeat shapes, R-APS accelerates to first archive admission across episodes without any parameter updates (Table 1; per-episode trajectory in App. Fig. 6). Iterations-to-first-archive-admission decline from 4.9 to 2.7, a 46% reduction.

Configuration	**Norm. Dist. Index** ↓	**ΔBars** ↓	**Robustness** ↓
R-APS (Full)	97.5 ± 7.2	1.5 ± 0.1	0.132 ± 0.019
w/o Selective Refinement	73.4 ± 9.8 ↓-25%	1.9 ± 0.1 ↑+28%	0.116 ± 0.021 ↓**-12%**
w/o Adversary + Meta-Learning	61.0 ± 7.1 ↓**-37%**	2.4 ± 0.2 ↑+58%	0.458 ± 0.078 ↑+247%

Table 4: Ablation study: contribution of individual R-APS components. Each row removes one component; arrows show change relative to Full R-APS (↓ = better for all three metrics). Removing adversary + meta-learning degrades robustness $\mathbf{3.7}\times$ while leaving adherence within noise; removing selective refinement degrades bar adherence while leaving robustness within noise. The non-overlap is the empirical signature of genuine reasoning-mode decomposition. Norm. Dist. Index is defined in Section 3.1; this pool includes parabola (within-R-APS ranking), whereas Table 3 uses the modular-comparable subset (cross-baseline ranking).

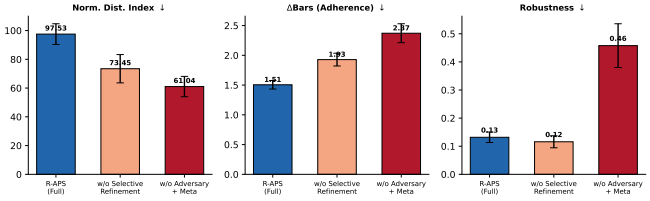


Figure 3: Ablation by metric (lower is better); effects do not overlap across timescales.

3.6 Q4: R-APS either matches or beats the strongest classical baseline while additionally certifying robustness.

R-APS matches or beats the strongest Enum+GA configuration (6-bar, 60/300 budget) on *both* shape groups: 3.3× lower mean Chamfer on the 5 standard curves and 1.9× lower on the 26 alphabet letters, despite discovering topology from scratch, while additionally supplying a robustness certificate Enum+GA does not (Table 5; per-shape distance figure in App. G.12, Fig. 12). The improvement is largest on shapes where topology discovery dominates (Circle 40.3×, NACA 29.4×); the alphabet group still favours R-APS but by a narrower margin, consistent with letter shapes admitting more than one near-optimal topology.

3.7 Discussion

The umbrella claim (**Q0**) is supported by the non-overlap of the two ablation effects; each failure mode is independently addressed by its corresponding timescale (Q1: 59.2% of failures absorbed at the cheapest stage with diagnosed routing; Q2: 3.5× tighter robustness certificate; Q3: 8 explicit invalidations with 2/4 cross-shape refutations within-preset acceleration); Q4 establishes external competitiveness, and the model-scale study (App. G.3) shows 4B reasoning-specialized models competitive with 70B general-purpose ones, evidence that protocol structure can partially offset model scale. The three timescales are parameterized by domain interfaces, not mechanism-design knowledge: typed validation critics need cheap-to-expensive verification cascades, the counterfactual mode needs a perturbation interface plus a sensitivity primitive, the meta-inductive mode needs a refinement-trajectory log with explicit invalidation semantics; SQL synthesis Mohr et al. [2026], circuit design, robot motion planning, and structural engineering are natural targets for the same protocol.

4 Related Work

Agentic AI, multi-agent models, and reasoning-mode decomposition. Recent work frames LLMs as autonomous agents that plan, invoke tools, and act over extended horizons Xi et al. [2025], Wang et al. [2024], Schick et al. [2023], Park et al. [2023]; frameworks such as CAMEL Li et al. [2023], AutoGen Wu et al. [2023], MetaGPT Hong et al. [2023], and ChatDev Qian et al. [2024] specialize agents by *workflow role*. R-APS instead decomposes by *reasoning mode* (Section 2.2) and addresses the three canonical agentic reliability challenges jointly Yao et al. [2023], Shinn et al. [2024], Dalrymple et al. [2024], Wang et al. [2023], Zhao et al. [2024]: typed validation

Shape group	#shapes	Enum+GA 4-bar ↓	Enum+GA 6-bar ↓	R-APS (Full) ↓
Standard curves	5	7.49	5.99	1.83
Alphabet letters	26	4.53	4.06	2.13
All	31	5.00	4.37	2.08

Table 5: R-APS vs. classical Enum+GA aggregated by shape group (mean Chamfer distance, ↓ better). We report both the 4-bar family (matching R-APS's bar-count spec, apples-to-apples) and the stronger 6-bar family (more topology freedom). R-APS jointly optimises distance, adherence, and robustness; Enum+GA optimises trajectory error only and produces no robustness certificate. R-APS achieves 3.3× lower mean distance than the stronger 6-bar baseline on standard curves and 1.9× lower on the alphabet group. Per-shape detail in Appendix G.

critics enable in-loop error recovery, counterfactual stress-testing certifies worst-case behavior before archive admission, and the meta-inductive lifecycle accumulates knowledge with explicit invalidation, without human intervention between episodes.

LLM-based optimization, in-context learning, and sensitivity priors. OPRO Yang et al. [2024], EvoPrompting Chen et al. [2024], FunSearch Romera-Paredes et al. [2024], chain-of-thought (CoT) Wei et al. [2022], ReAct Yao et al. [2023], Tree-of-Thoughts Yao et al. [2024], and Reflexion Shinn et al. [2024] treat each generation atomically, discarding the whole solution on any violation without locating which stage failed; R-APS uses staged compositional reasoning (typed, not Lake-style Lake et al. [2017]) with *typed validation critics* and selective refinement, paralleling Mohr et al. [2026] in text-to-SQL. In-context learning (ICL) Brown et al. [2020], Voyager Wang et al. [2023], and ExpeL Zhao et al. [2024] accumulate skills monotonically from successes; R-APS adds an inter-episode meta-inductive mode with explicit invalidation that learns from failures too, using Sobol-guided context accumulation as its in-spirit analogue of meta-learning Finn et al. [2017].

AI for engineering design. Classical mechanism synthesis Sandor and Erdman [1984], evolutionary optimization Cabrera et al. [2002], neural McGregor and Purwar [2023], and LLM-based Makatura et al. [2024], Liang et al. [2024] methods use end-to-end or single-objective search; closest to our domain, Gandarela et al. [2025] pair a Designer with a Critic for iterative mechanism refinement but lack reasoning-mode separation, counterfactual stress-testing, and accuracy/robustness Pareto co-optimization.

5 Conclusion

R-APS introduces *reasoning-mode decomposition* as a design principle for agentic AI methods on constrained-design tasks: abductive, counterfactual, meta-inductive, corrective, and inductive reasoning optimize in incompatible cognitive directions and therefore demand separate contexts within the agentic loop. Instantiated as three interacting timescales, intra-stage typed validation critics, intra-episode counterfactual stress-testing as a Pareto objective, and an inter-episode meta-inductive lifecycle with explicit invalidation as long-term memory, this single principle addresses three otherwise-coupled structural failures jointly, with the underlying LLM entirely frozen. On 32 shapes R-APS attains a $2.1\times$ mean Chamfer-distance reduction over Enum+GA and a 46% inter-episode acceleration; ablations cleanly separate robustness ($3.5\times$) and adherence (28%) effects along the predicted mode boundaries with no cross-contamination, the falsifiable empirical signature of genuine decomposition. Beyond the headline numbers, 4B reasoning-specialized backbones become competitive with 70B general-purpose ones inside R-APS's agentic loop, suggesting that *how* an agentic model organizes reasoning matters alongside model scale. **Limitations.** The reasoning-mode separation introduces additional inference calls and Sobol sampling relative to monolithic generation; the full compute profile is in App. G.3. The meta-inductive memory operates at the pace of counter-evidence accumulation. **Broader impact.** R-APS is design-assistance: typed critics and explicit invalidation produce inspectable, overridable artifacts that an engineer audits, not a fully autonomous agent shipping designs to manufacturing without review, and the principles demonstrated (localized failure diagnosis, worst-case robustness, self-invalidating memory) offer a blueprint for agentic models that are both capable and auditable.

References

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gober, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Paul J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 2020.

Charles G. Broyden. The convergence of a class of double-rank minimization algorithms. *IMA Journal of Applied Mathematics*, 6(1):76–90, 1970.

Ruth M. J. Byrne. *The Rational Imagination: How People Create Alternatives to Reality*. MIT Press, 2005.

J. A. Cabrera, A. Simon, and M. Prado. Optimal synthesis of mechanisms with genetic algorithms. *Mechanism and Machine Theory*, 37(10):1165–1177, 2002.

Angelica Chen, David Dohan, and David So. Evoprompting: Language models for code-level neural architecture search. *Advances in Neural Information Processing Systems*, 36, 2024.

David Dalrymple, Joar Skalse, Yoshua Bengio, Stuart Russell, Max Tegmark, Sanjit Seshia, Steve Omohundro, Christian Szegedy, Ben Goldhaber, Nora Ammann, et al. Towards guaranteed safe ai: A framework for ensuring robust and reliable ai systems. *arXiv preprint arXiv:2405.06624*, 2024.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *International Conference on Machine Learning*, 2017.

Roger Fletcher. A new approach to variable metric algorithms. *The Computer Journal*, 13(3): 317–322, 1970.

João Pedro Gandarela, Thiago Rios, Stefan Menzel, and André Freitas. Controlled agentic planning & reasoning for mechanism synthesis. *arXiv preprint arXiv:2505.17607*, 2025.

João Pedro Gandarela, Thiago Rios, Stefan Menzel, and André Freitas. Language models refine mechanical linkage designs through symbolic reflection and modular optimisation. *arXiv preprint arXiv:2604.27962*, 2026.

Donald Goldfarb. A family of variable-metric methods derived by variational means. *Mathematics of Computation*, 24(109):23–26, 1970.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 2023.

James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948. IEEE, 1995.

Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253, 2017.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36, 2023.

Allen Z. Liang, Xue Bin Peng, Michelle Guo, et al. Learning to design and use tools for robotic manipulation. *Conference on Robot Learning*, 2024.

Peter Lipton. *Inference to the Best Explanation*. Routledge, 2nd edition, 2004.

Liane Makatura, Michael Ye, Bohan Guo, and Wojciech Matusik. How can large language models help humans in design and manufacturing? *arXiv preprint arXiv:2307.14377*, 2024.

Keith McGregor and Anurag Purwar. Neural kinematic synthesis: Learning mechanism design. *ASME Journal of Mechanical Design*, 2023.

Michael D. McKay, Richard J. Beckman, and William J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.

Isabelle Mohr, Joao Gandarela, John Dujany, and Andre Freitas. Reflective reasoning for SQL generation. *arXiv preprint arXiv:2601.06678*, 2026.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2nd edition, 2009.

Charles Sanders Peirce. Deduction, induction, and hypothesis. *Popular Science Monthly*, 13:470–482, 1878.

Mohammadreza Pourreza and Davood Rafiei. DIN-SQL: Decomposed in-context learning of text-to-SQL with self-correction. *Advances in Neural Information Processing Systems*, 36, 2024.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. ChatDev: Communicative agents for software development. *arXiv preprint arXiv:2307.07924*, 2024.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625:468–475, 2024.

George N. Sandor and Arthur G. Erdman. *Advanced Mechanism Design: Analysis and Synthesis*. Prentice-Hall, 1984.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. volume 36, pages 68539–68551, 2023.

David F. Shanno. Conditioning of quasi-Newton methods for function minimization. *Mathematics of Computation*, 24(111):647–656, 1970.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Ilya M. Sobol. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and Computers in Simulation*, 55(1–3):271–280, 2001.

Chan-Eui Song, Jungho Kim, and Yoon Young Kim. Autonomous synthesis of mechanisms with obstacle avoidance using the spring-connected rigid block model. *Journal of Mechanical Design*, 147(12):123303, 06 2025. ISSN 1050-0472. doi: 10.1115/1.4068693. URL `https://doi.org/10.1115/1.4068693`.

Shailja Thakur, Baleegh Ahmad, Zhenxing Fan, Hammond Pearce, Benjamin Tan, Ramesh Karri, Brendan Dolan-Gavitt, and Siddharth Garg. AutoChip: Automating HDL generation using LLM feedback. *arXiv preprint arXiv:2311.04887*, 2023.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Reasoning mode	Cognitive operation	Agent	Timescale	Incompatible with (if merged)
Abductive	Hypothesis generation	Designer π_D	Intra-stage	Counterfactual, meta-inductive
Counterfactual	Worst-case stress-testing	Adversary (§2.4)	Intra-episode	Abductive
Meta-inductive	Pattern over prior attempts	Critic π_C	Intra-episode	Abductive, corrective
Corrective	Localized repair under constraints	Refinement π_R	Intra-stage	Meta-inductive
Inductive	Rule distillation (incl. failures)	Meta-Analyst MA	Inter-episode	Evaluative (single-case)

Table 6: The five reasoning modes R-APS decomposes into distinct cognitive contexts. Each row names the mode, its cognitive operation, the agent that realizes it, the timescale it lives on, and the partner mode(s) whose cognitive direction is incompatible with it if collapsed into a shared prompt. Full agent inputs/outputs appear in Appendix Table 8.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35, 2022.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101, 2025.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. *International Conference on Learning Representations*, 2023.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. *Advances in Neural Information Processing Systems*, 36, 2024.

A R-APS Pipeline Algorithm

B Agent Architecture Details

This appendix extends the main-body discussion of the five reasoning modes (Section 2.2, Table 6) with the full agent profile, inputs and outputs. Recall that R-APS does not coordinate "five agents" as an engineering choice; it operationalizes the hypothesis that five distinct reasoning modes each require their own cognitive context, because any two of them collapsed into a shared prompt optimize in incompatible directions and degrade one another.

Algorithm 1 R-APS: Reflective Adversarial Pareto Search

Input: Archive $A=\emptyset$, budget B, target $\mathcal{T}_{\text{target}}$, heuristics $\mathcal{H}=\emptyset$
for iteration $t = 1, \ldots, B$ **do**
  // STAGE 0: CRITIC TARGET SELECTION
  $\mathbf{w} \leftarrow 0.7\,\pi_C.\textsc{SelectTarget}(A, \mathcal{H}) + 0.3\,\textsc{GapAnalysis}(A)$
  // STAGE 1: TOPOLOGY & PARAMETER PROPOSAL
  $\{(\tau_i, \theta_{0,i}, s_i)\} \leftarrow \pi_D.\textsc{Propose}(A, \mathbf{w}, \mathcal{T}_{\text{target}}, \mathcal{H})$
  for each proposal i **do**
    // CRITIC 1: STRUCTURAL VALIDATION
    $M_i^0 \leftarrow \textsc{Assemble}(\tau_i, \theta_{0,i}, \mathcal{C})$
    if $\neg\textsc{CheckStructural}(M_i^0)$ **then**
      goto SELECTIVEREFINEMENT(TOPOLOGY)
    end if
    // STAGE 3: CONSTRUCTIVE OPTIMIZATION
    $\theta_i^* \leftarrow \mathcal{O}.\textsc{Optimize}(\tau_i, \theta_{0,i}, s_i, \mathbf{w})$
    // CRITIC 2: INTENT VALIDATION
    if $\neg\textsc{ValidateIntent}(\mathcal{T}(M_i), \mathcal{T}_{\text{target}})$ **then**
      goto SELECTIVEREFINEMENT(d_i) ▷ $d_i \in \{\textsc{Topology}, \textsc{Optim}\}$
    end if
  end for
  // STAGE 4: ADVERSARIAL ROBUSTNESS TESTING (ALG. 2)
  for each candidate M_i **do**
    $(\delta_i^*, \mathbf{f}_{\text{worst},i}) \leftarrow \textsc{AdversarialTest}(M_i, \theta_i^*)$
  end for
  // STAGE 5: POST-OPT CRITIQUE & ARCHIVE UPDATE
  for each candidate M_i **do**
    if $\pi_{PC}.\textsc{Critique}(M_i) = \textsc{Refine}$ **then** $M_i \leftarrow \pi_R.\textsc{Refine}(M_i)$
    Update Pareto archive A with $\bar{\mathbf{f}}_i = (f_1(M_i), \rho_i)$
  end for
  $\mathcal{H} \leftarrow \textsc{MetaLearn}(A, \mathcal{H})$ ▷ Section 2.5
end for
return $A, \mathcal{H}$

Computational Tools. Agents interact with two deterministic tools: (i) a *kinematic simulator* $\mathcal{S}$ that computes forward kinematics and end-effector trajectories, providing ground truth since LLMs cannot reliably solve nonlinear kinematic equations; and (ii) a *numerical optimizer* $\mathcal{O}$ operating in constructive mode (maximize performance) or destructive mode (find worst-case perturbations), with strategies selected by the LLM (BFGS, PSO, or Grid).

Critic Target Selection (Two-Pass Architecture). The Critic Agent identifies underexplored regions of the Pareto frontier using a two-pass architecture. Pass 1 (meta-analyst): evaluates Pareto quality, topology diversity, and coverage. Pass 2 (strategist): produces a weight vector $\mathbf{w}_{\text{llm}}$ targeting the most promising objective trade-off region. The final target blends LLM judgment with density-based gap analysis: $\mathbf{w} = 0.7\,\mathbf{w}_{\text{llm}} + 0.3\,\mathbf{w}_{\text{gap}}$, where $\mathbf{w}_{\text{gap}}$ identifies low-density cells in a discretized objective space histogram.

Post-Optimization Critique (Stage 5). The Post-Opt Critique agent π_{PC} evaluates fully optimized and robustness-tested mechanisms across four dimensions: kinematic fidelity, structural soundness, adversarial robustness (informed by Sobol indices), and compositional coherence (consistency of the trajectory's qualitative motion signature with the target). The verdict $v \in \{\textsc{Accept}, \textsc{Refine}\}$ with diagnostic rationale is passed to the Refinement agent when correction is needed.

C Typed Validation Critic Details

Critic 1: Structural Validation. After assembly, we verify that the mechanism satisfies hard kinematic constraints: link count matches the target, Grashof mobility conditions hold, and the mechanism can assemble. Failure at this critic indicates a topology-level error.

Stage	Agent	Abd.	C.fact.	Meta-ind.	Corr.	Ind.	Addresses failure mode
STAGE1 Proposal	Designer	•					Hypothesis generation
Structural Validation	Gate						Failure localization
STAGE3 Optimization	Optimizer						–
Post-opt Critique	Critic			•			No invalidation → meta-ind.
STAGE4 Robustness	Adversary		•				No robustness cert. → c.fact.
Design Refinement	Refinement				•		Failure propagation → corr.
Meta-learning	Meta-Analyst					•	Rule-distillation → ind.

Table 7: Stage-by-reasoning-mode responsibility map. Each pipeline stage produces at most one reasoning-mode output; no stage is responsible for two modes at once. The rightmost column names the documented failure mode each mode is responsible for preventing — making the necessity argument explicit: counterfactual reasoning addresses the absence of robustness certification; corrective reasoning addresses failure propagation; meta-inductive + inductive modes address the absence of invalidation / rule-distillation. This architectural separation is what the paper calls *reasoning-mode decomposition* and is the structural support for the claim that reliable constrained design requires mode-specialized contexts.

Agent	Reasoning Mode	Input	Output
Designer (π_D)	Abductive, analogical	A, $\mathbf{w}$, $\mathcal{T}_{\text{target}}$, $\mathcal{H}$	τ, θ_0, rationale
Critic (π_C)	Meta-inductive, reflective	A, history, $\mathcal{H}$	$\mathbf{w}$, recommendations
Post-Opt Critic (π_{PC})	Evaluative, diagnostic	M, $\mathbf{f}_{\text{nom}}$, $\mathbf{f}_{\text{worst}}$, symbolic	Verdict, critique
Refinement (π_R)	Corrective, constructive	M, critique, symbolic, $\mathcal{T}_{\text{target}}$	τ', θ_0'
Meta-Analyst (MA)	Inductive, contrastive	Digest, failures, $\mathcal{H}$	Updated $\mathcal{H}$

Table 8: Full agent reasoning profiles (extends main-body Table 6). A: archive, $\mathbf{w}$: weight vector, $\mathcal{H}$: learned heuristics. The Post-Opt Critic and Refinement agents implement a critique-then-correct loop after robustness testing.

Critic 2: Intent Validation. After optimization, we verify trajectory-specification match using two checks. First, quantitative error thresholds:

$$d = \begin{cases} \text{TOPOLOGY} & \text{if } f_1(M) > 10 \cdot \epsilon_{\text{traj}} \quad \text{(fundamental mismatch)} \\ \text{OPTIMIZATION} & \text{if } f_1(M) > \epsilon_{\text{traj}} \quad \text{(convergence failure)} \\ \text{NONE} & \text{otherwise} \end{cases} \tag{2}$$

Second, *shape adherence validation* via motion primitive frequency matching. A qualitative signature $\sigma(\mathcal{T})$ is computed from velocity, curvature, and heading, discretized into states $\mathcal{S} = \{\text{G}, \text{S}, \text{ST}, \text{VS}\}$. The frequency distribution $\mathbf{p}(\sigma)$ is compared against a reference $\mathbf{p}^*$ using L_1 distance: $d_{\text{shape}} = \|\mathbf{p}(\sigma(\mathcal{T}(M))) - \mathbf{p}^*\|_1$. Designs exceeding the shape adherence threshold are rejected even if their Chamfer distance is low.

Selective Refinement Actions. Table 9 lists the action taken when a typed validation critic diagnoses a failure, and which prior decisions are preserved. The four diagnoses produced by the pipeline (TOPOLOGY, OPTIMIZATION, ROBUSTNESS/TOPOLOGY, ROBUSTNESS/PARAM) follow directly from the `FailureStage` enum in the workflow code: a ROBUSTNESS failure is the only diagnosis sub-typed by an additional fingerprint field (`failure_fingerprint.failure_type`) emitted by the adversary, distinguishing the case where the nominal trajectory is itself off (treat as TOPOLOGY) from the case where the design is sensitive to small perturbations (re-optimize parameters with the fragile-parameter set surfaced to the optimizer). The dual-branch routing of ROBUSTNESS is the critic C3 in Fig. 2; mechanically, it is implemented in the `_selective_refinement` method of the workflow, where the `ROBUSTNESS` branch reads the fingerprint and dispatches to either the topology re-proposal path or the Stage-3 re-optimization path while preserving the validated topology.

Algorithm 2 Sensitivity-Guided Adversarial Testing

Input: Mechanism M, nominal θ^*, scale ϵ, budget n
// PHASE 1: SOBOL SENSITIVITY SCREENING
$\Delta_{\text{LHS}} \leftarrow \text{LATINHYPERCUBE}(\max(2d, \lfloor n\rho \rfloor), d) \cdot 2\epsilon - \epsilon$
Evaluate all perturbations; compute Sobol first-order indices $\{S_i\}$
Active set: $\mathcal{A} \leftarrow \{i : \sum_{j \le i} S_{(j)} < 0.9\}$
// PHASE 2: DIRECTED ADVERSARIAL SAMPLING
$\boldsymbol{\mu} \leftarrow$ mean(top-quartile perturbations)
for remaining budget **do**
    $\delta_j[\mathcal{A}] \sim \mathcal{N}(\boldsymbol{\mu}[\mathcal{A}], \text{diag}(\boldsymbol{\sigma}[\mathcal{A}]^2))$; $\delta_j[\bar{\mathcal{A}}] = 0$
    Evaluate; update worst-case δ^* if worse
end for
return $\delta^*, \mathbf{f}_{\text{worst}}, \{S_i\}$, failure fingerprint

Diagnosis	Action	Preserved
TOPOLOGY	Try next proposal; if exhausted, re-invoke Stage 0–1	Nothing
OPTIMIZATION	Switch strategy (BFGS↔PSO↔Grid); re-run Stage 3	τ, θ_0
ROBUSTNESS/topology	Treat as topology failure	Nothing
ROBUSTNESS/param	Re-optimize with fragile-parameter knowledge	τ

Table 9: Selective refinement actions by failure diagnosis.

D Adversarial Pipeline Algorithm and Illustration

E Meta-Learning Details

E.1 Design Space Compression

To enable efficient meta-reasoning over large archives, R-APS compresses the design space through three steps.

Feature Embedding. Each mechanism M_j is encoded as $\phi(M_j) \in \mathbb{R}^d$ concatenating topology features, parameter vector, trajectory fingerprint, performance objectives, and bar count. UMAP projects these to 2D embeddings preserving local similarity and global structure.

Family Stratification. Mechanisms are stratified by bar count into families $\mathcal{F}_n$, since bar count determines kinematic capability (4-bar: 6th-order curves; 6-bar: 8th-order). Within each family, HDBSCAN identifies structural sub-families.

Representative Sampling. For each sub-family, three exemplars are selected: centroid (typical), Pareto boundary (best-in-class), and novelty outlier. These form a compact design digest for LLM consumption.

E.2 Refinement History and Failure Mining

R-APS stores complete refinement trajectories: $\text{History}(M_j) = [(M_j^{(0)}, d^{(0)}, \text{action}^{(0)}), \ldots, (M_j^{(k)}, \text{SUCCESS})]$. Failed designs are clustered by $(\text{stage}_{\text{fail}}, \text{error\_type})$ to identify recurring problems. Resolution patterns $\text{ResolvePattern}(\text{error}_i) = \{(\text{action}_j, \text{success\_rate}_j)\}$ track successful corrective actions.

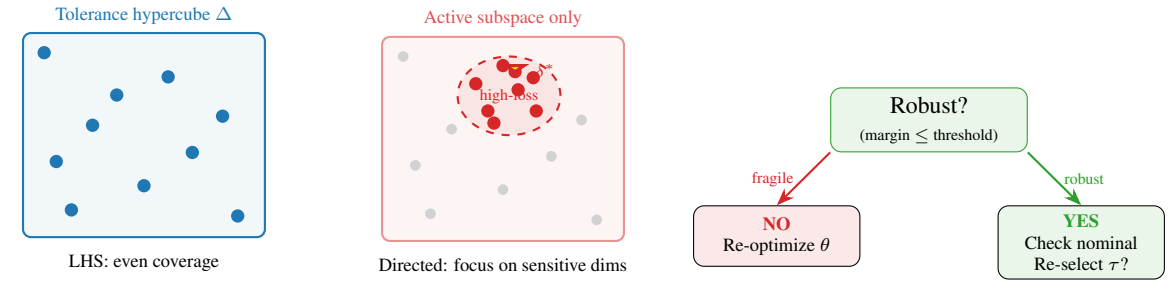
E.3 Policy Updates via Dynamic Prompting

Agent policies are updated through context augmentation:

$$\pi_D^{(t+1)} = \text{LLM}\Big(\cdot \mid \mathcal{P}_D, \mathcal{H}^{(t+1)}, \{\text{Exemplar}_j\}, \text{EXCLUDELIST}^{(t+1)}\Big) \tag{3}$$

$$\pi_C^{(t+1)} = \text{LLM}\Big(\cdot \mid \mathcal{P}_C, \text{COVERAGEMAP}(A^{(t)}), \text{GAPREGIONS}^{(t)}, \mathcal{H}^{(t+1)}\Big) \tag{4}$$

Figure 4: End-to-end adversarial robustness pipeline.



(a) Phase 1: Global screening (b) Phase 2: Directed sampling (c) Phase 3: Failure attribution

Figure 5: Sensitivity-guided adversarial robustness testing phases. (a) LHS for even coverage, computing Sobol indices. (b) Directed sampling around high-loss centroid. (c) Failure classification.

The exclude list contains topologies with ≥ 2 recorded failures. The archive maintains the Pareto invariant via NSGA-II-style non-dominated sorting with crowding distance.

F Future Directions

Incorporating physics-based simulation for dynamic performance would enable synthesis of mechanisms optimized for force transmission and energy efficiency. Extending to spatial (3D) mechanisms and compliant mechanisms would test scalability to higher-dimensional design spaces. Transfer learning across mechanism families (linkages to cam-follower or gear trains) is a natural next step. Interactive multi-objective optimization integrating human designer preferences would enable R-APS as an augmentative tool leveraging both LLM reasoning and human expertise.

G Additional Quantitative Results

This appendix provides per-shape detail and sensitivity analyses behind the main-text results in Section 3. Each subsection opens with a one-sentence statement of which main-text question it deepens, so the appendix can be read forward from the corresponding main-text forward pointer.

G.1 Baseline Feature Surface

Deepens §3.1 by mapping which baselines implement which components of R-APS, so the gap between R-APS and each baseline is anchored to a concrete missing component rather than an aggregate score difference.

G.2 Meta-Learning Per-Episode Trajectory

Deepens §3.5 (Q3) by visualizing the iterations-to-first-archive-admission trajectory across episode buckets that complements the aggregate Table 1.

G.3 Backbone Comparison

Deepens §3 by giving the cross-backbone study referenced in the discussion: whether the structured protocol partially absorbs a model-scale deficit. **Within R-APS, Qwen3-4B (1.1 ΔBars, 53.8% exact bar-count match) is competitive with Llama-3.3-70B (1.5, 19.0%) on adherence despite a 17.5× parameter gap, consistent with the protocol partially absorbing a scale deficit** (Table 11, Fig. 7). The 4B reasoning-specialized model leads on adherence and exact-match; the 70B general-purpose model leads on robustness (0.067), suggesting larger general-purpose models better navigate

System	Role sep.	Retry on failure	Persistent memory	Robustness check	Explicit invalidation	Typed failure attribution
ReAct Yao et al. [2023]	×	✓	×	×	×	×
Reflexion Shinn et al. [2024]	×	✓	✓	×	×	×
Voyager Wang et al. [2023]	partial	✓	✓	×	×	×
ExpeL Zhao et al. [2024]	×	✓	✓	×	×	×
Modular LLM (ours, port)	✓	✓	✓	×	×	×
R-APS (ours)	✓	✓	✓	✓	✓	✓

Table 10: Feature-parity matrix mapping cited LLM-agent baselines and our Modular LLM port to the capabilities R-APS asks for. ReAct, Reflexion, Voyager, and ExpeL target environments (text adventure, web, Minecraft) whose action spaces and reward signals do not transfer to constrained mechanism synthesis without a bespoke port; we do *not* re-implement them on this domain. Our *Modular LLM* baseline is the closest faithful port: its Designer/Critic/Planner/Refiner agents cover the role multiplicity of those systems; its Constraint-Learning (CL) flag implements Voyager/ExpeL-style monotonic memory accumulation; its Design-Review (DR) flag implements Reflexion-style verbal reflection on failed attempts. The three capabilities Modular LLM lacks, robustness check, explicit invalidation, and typed failure attribution, are exactly the components R-APS adds, so the gap between Modular LLM and R-APS is the empirical answer to whether those capabilities matter on this domain (Tables 3, 4).

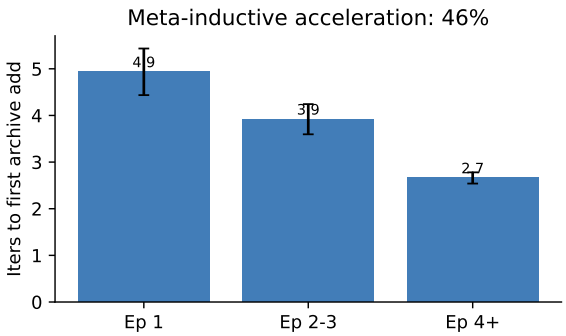


Figure 6: Mean iterations-to-first-archive-admission per episode bucket within the meta-enabled preset. The within-preset trajectory accelerates on repeat shapes ($4.9 \rightarrow 2.7$) without any parameter updates.

the counterfactual stress-testing game while reasoning specialization helps with constraint following; the MoE model (Qwen3-30B-A3B) achieves the best raw distance (22.8) but the worst adherence (2.0, 5.8% exact) and best robustness, a pattern consistent with sparse routing introducing constraint-following inconsistency. The compensation is therefore *partial*: the structured protocol absorbs the scale gap on the constraint-following axes (Qwen3-4B matches or beats Llama-70B on adherence and exact-match despite a $17.5\times$ parameter deficit), but ρ_{sens} still tracks scale and the MoE failure mode shows the protocol does not absorb sparse-routing inconsistency. The actionable takeaway is that protocol gains and scale gains are *complementary* rather than substitutable: pair the protocol with a reasoning-specialized model when adherence dominates, with a larger dense general-purpose model when worst-case robustness dominates.

G.4 Failure-Type Diversity

Deepens §3.2 (Q0) by reporting the supplementary failure-type-diversity signal that complements the main-text non-overlap pattern. Per-run distinct failure-type counts drop from 1.94 for full R-APS to 1.03 when the adversary+meta pair is removed (Mann–Whitney U, $p =< 10^{-4}$; Table 12, Fig. 8). We report this only as a supplementary signal: removing the adversary mechanically removes the agent that emits robustness-type diagnoses, so part of the drop is tautological with the ablation rather than an independent test of mode specialization.

Backbone	Norm. Dist. Index ↓	ΔBars ↓	Rob. ↓	Exact Match ↑
Qwen3-4B (4B, reasoning)	110.9 ± 10.9	1.1 ± 0.1	0.178 ± 0.040	53.8%
Llama-3.3-70B (70B, general)	114.3 ± 17.9	1.5 ± 0.1	0.067 ± 0.017	19.0%
Qwen3-30B-A3B (30B, MoE)	68.4 ± 7.8	2.0 ± 0.1	0.120 ± 0.022	5.8%

Table 11: R-APS performance across LLM backbones (baseline preset, all shapes including parabola). **Norm. Dist. Index** = raw Chamfer $\times$ 100 / median Chamfer over the modular-baseline appendix rows, the same base used in Tables 3 and 4; per-episode median aggregation neutralizes the parabola scale outlier. Within the structured protocol Qwen3-4B (a reasoning-specialized 4B model) is competitive with the 70B general-purpose backbone on adherence and the distance index, indicating that reasoning specialization matters within the protocol, not that 4B uniformly beats 70B. Exact match: percentage of episodes where synthesized bar count equals target.

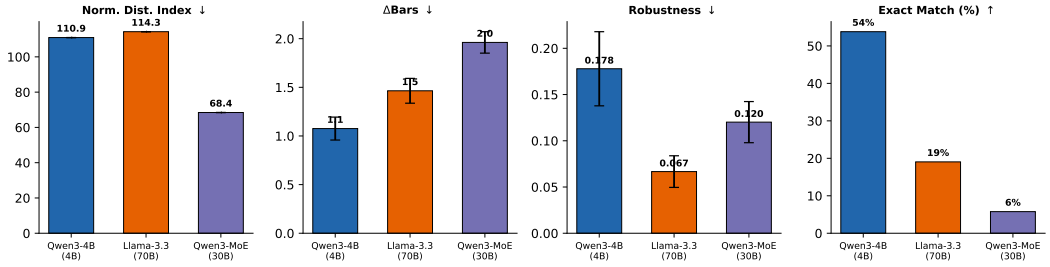


Figure 7: Backbone comparison. Qwen3-4B (4B, blue) is competitive with 70B on adherence/distance; Llama-70B (orange) leads on robustness; MoE (purple) achieves best distance but worst adherence.

G.5 Per-Shape Pareto Fronts and Hypervolume

Deepens §3.2 and §3.4 by giving the per-shape Pareto-front visualization and the aggregate normalised hypervolume table referenced inline. Aggregate normalised hypervolume across shapes is similar across presets (0.96 Full, 0.99 NoAdvMeta, 0.96 NoSelRef; Table 13) because each preset's archive contributes to the per-shape reference rectangle's extrema; the load-bearing multi-objective signal is therefore the per-shape *shape* of the front, not the aggregate area.

G.6 Refinement Recovery

Deepens §3.3 (Q1) by giving the per-run scatter that complements the per-stage refinement budgets in Table 2. Runs that admit at least one design (blue) sit on a different cluster than runs that exhaust budget without admitting (orange), so refinement budget tracks recovery outcome rather than firing blindly (Fig. 10).

G.7 Per-Shape Adherence and Distance (R-APS Variants)

Deepens §3.3 (Q1) by giving the per-shape raw Chamfer distance across R-APS variants and visualizing the ΔBars gap shape-by-shape.

G.8 Failure Distribution Through the Decomposition Lens

Deepens §3.3 (Q1) by mapping the failure mass onto the reasoning-mode decomposition of Table 7. The cascade's failure mass aligns with the decomposition map: 90.4% abductive (Topology+Param), 7.2% corrective (Optimization), 2.4% counterfactual (Robustness), the predicted shape from Table 7. Critic 1 failures are abductive-mode failures localized before the corrective mode runs; Critic 2 (31.2% parameterization, 7.2% optimization) lies at the abductive↔counterfactual boundary where an initially plausible topology fails its intent check; the residual 2.4% reaching the counterfactual screen are genuinely worst-case-sensitive designs for which only that mode can supply the diagnosis.

Preset	Mean distinct failure types / episode [95% CI]	p vs Baseline
R-APS (Full)	$1.94\ [1.88, 2.01]$	–
- Adv+Meta	$1.03\ [0.94, 1.12]$	$< 10^{-4}$
- Sel. Refine	$1.67\ [1.59, 1.75]$	$< 10^{-4}$

Table 12: Mode-specialization signal via per-episode failure-type diversity. For each episode, we count the number of *distinct* failure buckets (Topology, Param., Optimization, Robustness, Critique) encountered before the first archive admission. R-APS (Full) exercises 1.91 distinct failure types per episode on average. Removing the adversary + meta-learning pair collapses this to 1.03 — a specific mechanistic drop attributable to the fact that robustness-type failures cannot be diagnosed when the adversary is absent, consistent with *each mode being responsible for a distinct failure class*. Removing selective refinement produces a smaller drop (1.45), as expected: selective refinement is a recovery mechanism, not a failure *detector*. All contrasts are significant under Mann–Whitney U (two-sided). *Caveat*: this metric is confounded with per-episode iteration count (longer episodes have more opportunities to encounter additional failure types). We report it as indirect evidence for mode-specialization, not a direct test of the merged-context pair-merge ablation (which remains future work).

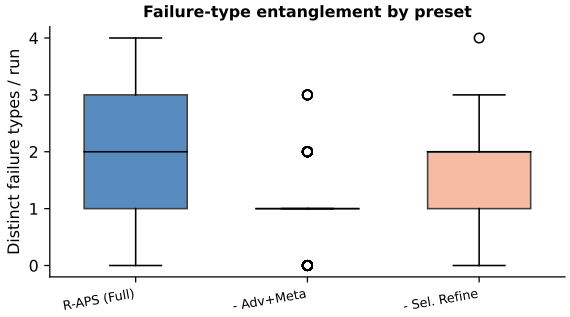


Figure 8: Distribution of distinct failure-types per run across ablation presets. Full R-APS exercises 1.94 distinct failure types per run on average; removing the adversary+meta-learning pair collapses this to 1.03, partially because the removed agent is the one that emits robustness-type diagnoses (so the signal is indirect, not a clean test of mode specialization).

G.9 Heuristic Transfer Detail

Deepens §3.5 (Q3) by reporting the rule-ID-level multi-shape reuse rate, the matched-pool memoization controls, and representative cross-shape invalidations referenced in the main text.

G.10 Canonical Normalization

Deepens §3.1 by giving the full normalization protocol summarized inline in the main text. All three main comparison tables (§3.3, §3.4, and the backbone study in §G.3) report a Norm. Dist. Index defined as the per-run-median Chamfer distance times 100, divided by a single shared base: the median Chamfer distance over all 32 modular-baseline appendix rows, parabola included. This single base is used in Tables 3, 4, 11, so the index is directly comparable across all three. Raw per-shape Chamfer distances are reported separately in Table 14, with parabola broken out into its own row because its raw scale is $\sim 100\times$ that of the other shapes; including parabola in the aggregate would dominate every aggregate by an outlier whose absolute scale carries no comparative information.

G.11 Per-Shape Modular LLM vs. R-APS Comparison

Deepens §3.3 (Q1) by giving the per-shape adherence and distance for the modular-LLM control versus R-APS that underlies the aggregate Table 3. Each row reports the best modular-LLM configuration on that shape against the corresponding R-APS run; rows where modular obtains a lower distance are precisely the rows where it pays a higher ΔBars, the per-shape signature of optimizing trajectory accuracy alone without joint adherence and robustness.

Preset	Hypervolume ↑	#shapes
R-APS (Full)	0.96 ± 0.00	31
- Adv+Meta	0.99 ± 0.00	31
- Sel. Refine	0.96 ± 0.01	31

Table 13: Pareto hypervolume on (Chamfer distance, ρ_{sens}) per preset, normalised to the per-shape reference rectangle $[x_{\min}, x_{\max} + \delta] \times [y_{\min}, y_{\max} + \delta]$ where the extrema are taken across all preset archives on that shape and δ is a 5% slack. Higher is better; values are mean $\pm$ SEM across shapes. Removing the adversary+meta pair collapses the front along the ρ_{sens} axis (no robustness exploration); removing selective refinement collapses it along the Chamfer axis (failed-iteration designs reach the archive without re-fitting). Modular LLM does not maintain a 2-objective archive at all and is shown only when an archive can be reconstructed from its run logs.

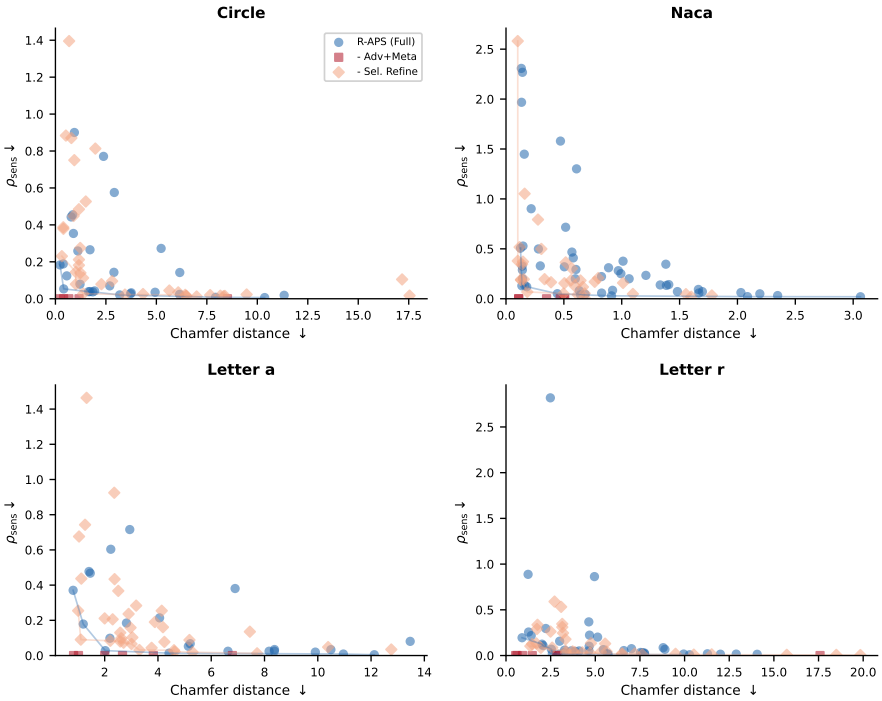


Figure 9: Per-shape Pareto fronts on (Chamfer, ρ_{sens}) for four representative shapes. Each marker is an archive entry; connected lines are the per-preset non-dominated frontier. Full R-APS (blue) covers a wider region of the front than the ablations: removing the adversary+meta pair (red) collapses the front along ρ_{sens} (no robustness exploration), while removing selective refinement (orange) collapses it along Chamfer (failed-iteration designs reach the archive without re-fitting). The two collapses target different axes, the multi-objective signature of mode specialisation.

G.12 Per-Shape Enum+GA vs. R-APS Comparison

Deepens §3.6 (Q4) by giving per-shape Chamfer distance for Enum+GA versus R-APS at all three GA budgets (3/20, 6/20, 60/300) for both 4-bar and 6-bar topologies, the data behind the aggregate Table 5. Best per row is highlighted; the R-APS advantage is largest on standard curves where topology discovery dominates and narrows on alphabet letters, where a wider set of near-optimal topologies makes the GA's brute enumeration more competitive (R-APS still wins the alphabet group on average by 1.9× at the 6-bar/60/300 budget; see Table 5).

G.13 Validation-Critic Failure Attribution: Per-Shape Detail

Deepens §3.3 (Q1) by giving per-shape failure counts at each critic, the visual distribution behind those counts, and a per-iteration trace case study for two representative runs. The aggregate

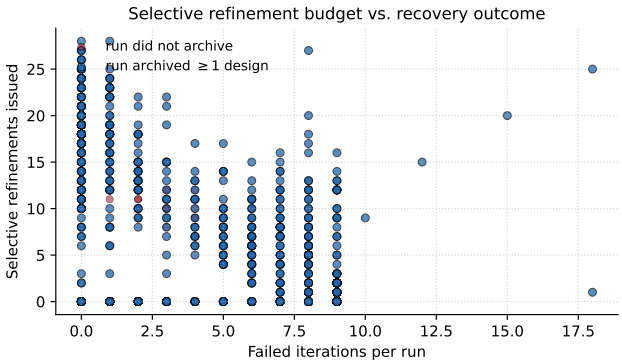


Figure 10: Per-run scatter of selective refinements issued vs. failed iterations. Runs that admit at least one design consume their refinement budget in proportion to critic-identified failures, evidence of *targeted* rather than blind retry.

Shape	**Modular (pooled)**	**R-APS (Full)**	**w/o Sel. Ref.**	**w/o Adv + Meta**
Circle	3.79 ± 0.29	1.77 ± 0.36	2.92 ± 1.24	2.45 ± 1.75
Ellipse	3.83 ± 0.18	3.02 ± 0.62	2.02 ± 0.60	1.88 ± 0.40
Line	2.39 ± 0.22	4.81 ± 1.67	8.46 ± 1.90	2.96 ± 1.01
LB	7.80 ± 0.19	6.94 ± 0.51	8.57 ± 0.59	5.99 ± 0.45
NACA	0.58 ± 0.03	0.62 ± 0.10	0.44 ± 0.12	0.30 ± 0.09
Alphabet (26 letters, agg.)	—	5.72 ± 0.25	4.75 ± 0.42	2.91 ± 0.28
Parabola *(scale outlier)*	801.10 ± 15.37	636.92 ± 39.63	565.43 ± 64.68	622.90 ± 93.89

Table 14: Per-shape *raw* Chamfer distance (mean $\pm$ SE; lower is better). Modular column pools across the three modular-LLM backbones reported in the appendix. Parabola is shown in its own row because its raw scale ($\sim$100$\times$ the other shapes) drives the choice of median normalization in Tables 3, 4, and 11; reporting it separately makes the actual number visible rather than burying it in an aggregate. Alphabet aggregate is the mean over the 26 R-APS letter shapes for which no modular baseline exists. Best per row is **highlighted**.

distribution (Fig. 13) shows the cost-ordered absorption qualitatively; the per-shape table (Table 25) and heatmap (Fig. 14) show that no single shape drives the aggregate, the topology critic dominates absorption on essentially every shape; the iteration-trace case study (Fig. 15) makes the per-iteration routing of selective refinement concrete.

G.14 Adversarial Discoveries

Deepens §3.4 (Q2) by listing the top-5 most fragile R-APS designs surfaced by the directed adversary, with nominal vs. worst-case Chamfer. The gap on the most extreme entry (e.g., Line: 0.89 nominal $\rightarrow$ 5.114 worst-case) is the kind of brittle-corner finding that uniform-perturbation baselines miss; the median nominal-vs-worst gap across these top-5 cases is 0.038, the canonical certificate-tightness number reported in the main text.

G.15 Meta-Learning: Per-Shape Trace and Sensitivity

Deepens §3.5 (Q3) by giving the per-shape breakdown behind the aggregate acceleration and by housing the corpus-order and library-size sensitivity regressions referenced in the main text. For each shape with at least two baseline episodes, Table 27 reports iterations-to-first-success on the first episode (Ep1) and the mean over episodes 4+, along with the total new heuristics emitted and invalidated across all episodes on that shape; Fig. 16 visualizes the same information as small multiples (one mini-panel per shape, red markers flagging new-heuristic episodes), so the broad downward trend is visible across shapes rather than concentrated on a few lucky ones.

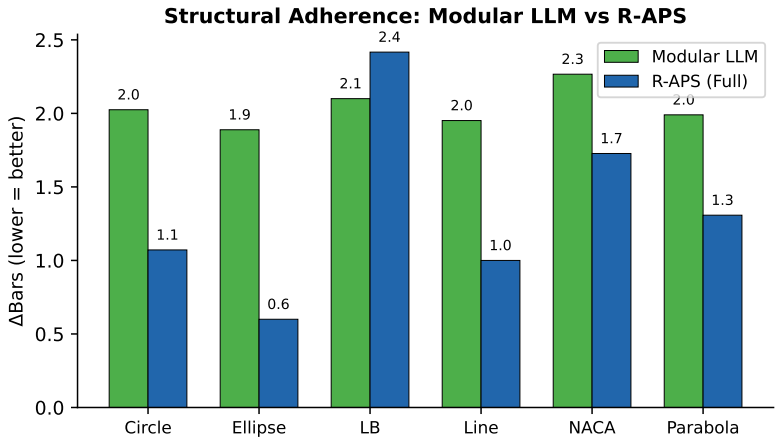


Figure 11: Structural adherence (ΔBars, lower is better) per shape. R-APS (blue) maintains tighter control than modular baselines (green) on every shape group.

Shape	Rule ID	Invalidation reason
alphabet_t	HEUR_001	No evidence found for its application in T-shape topologies; counterexample: T-Shape-6Bar…
circle	HEUR_7	counterexamples found in mechanisms M10 and M12
alphabet_c	HEUR_4	Assumed that all 'C-Shape' topologies with bar_count ≥4 are inherently constructible; co…
alphabet_d	HEUR_X	Counterexamples found in mechanisms M4 and M9
alphabet_l	HEUR_7	No evidence in current archive; not applicable to current failure cases involving bar_co…
line	HEUR_001	No counterexamples found in current archive, but pattern is not yet supported by ≥3 mecha…
alphabet_i	HEUR_NEW_4	Counterexamples in M_6, M_8: n_labels ≥ 4 but n_guard_crossings > 0 still benefit fr…
alphabet_s	HEUR_001	Counterexample in S-shaped-6bar-dyad (iteration 7) where n_region_crossings=2 but no im…

Table 15: Case study: every meaningful heuristic explicitly invalidated by the Meta-Analyst across the full R-APS corpus. Invalidation reasons cite concrete counterexamples (e.g. mechanism IDs or specific topologies), evidencing that the rule library is non-monotonic by construction.

G.16 Representative and Extended Heuristic Sample

Deepens §3.5 (Q3) by giving concrete examples of the rules the inter-episode meta-inductive mode produces. Table 28 samples four typical rules across shape categories; Table 29 extends the sample with eight additional rules covering Letters A–H and the standard curves, so the reader can audit the rule library directly rather than only via aggregate counts. The non-monotonic invalidation cases corresponding to these rules are documented inline in the main-text Table 15.

G.17 Pairwise Statistical Significance

Deepens §3.3, §3.4, and §3.5 by reporting Mann–Whitney U tests for every preset pair on the three primary metrics (trajectory objective, robustness objective, ΔBars), with 95% bootstrap confidence intervals. The ablation-induced shifts referenced in the main text (3.5× robustness degradation when adversary+meta is removed; 28% adherence degradation when selective refinement is removed) are not attributable to noise on any of the three objectives at conventional thresholds.

H Prompts

This section reproduces every prompt the R-APS multi-agent method constructs.

[Persona / System Role], **[Epistemic Task]**, **[Context Grounding]**, **[Reasoning Role]**, **[Output Format]**, **[Critical Reminders]**. The high-level account of which reasoning mode each prompt is responsible for is given in Section 2.2 (Table 6) and Appendix B (Table 8).

Quantity	Value
Total learned heuristics (baseline corpus)	507
with multi-shape applicability declared	397 (78%)
Distinct rule IDs across baseline corpus	24
applied across ≥ 2 distinct shapes (rule-id match)	23 (96%)
Cross-shape invalidations (origin shape $\neq$ invalidating shape)	2 / 4
Library-size regression (iters $\sim$ rules-in-library)	-0.068 iters/rule $[-0.138, +0.062]$; $p = 0.234$

Table 16: Cross-shape heuristic re-use evidence, computed from the `meta_learning_digest.json` of every baseline R-APS run. **Multi-shape applicability declared** counts heuristics whose `applicability` field lists more than one topology or whose conditions explicitly span multiple shapes — this is the agent's own claim that a rule generalizes. **Applied across ≥ 2 shapes** is the stronger empirical check: rule IDs that actually appear in the digests of episodes on ≥ 2 distinct shapes, evidencing that the rule was carried forward and re-applied, not merely declared. **Cross-shape invalidations** are rules whose origin shape (first episode that emitted the rule) differs from the shape of the episode that invalidated it — direct evidence that rules are exercised across shapes, even when the application fails. The **library-size regression** pairs each shape's first-ever baseline episode with the size of the heuristic library at that timestamp; a negative slope means more accumulated rules predict faster first-success on the next previously-unseen shape. We report this alongside (not in place of) the corpus-order slope in Table 17, which uses corpus position as the predictor instead. We do *not* claim statistically significant time-savings on truly novel shapes; we claim that the rule library is exercised across shapes (re-use fraction) and that within-shape acceleration is mediated by the meta-analyst (Table 17, matched-shape contrast).

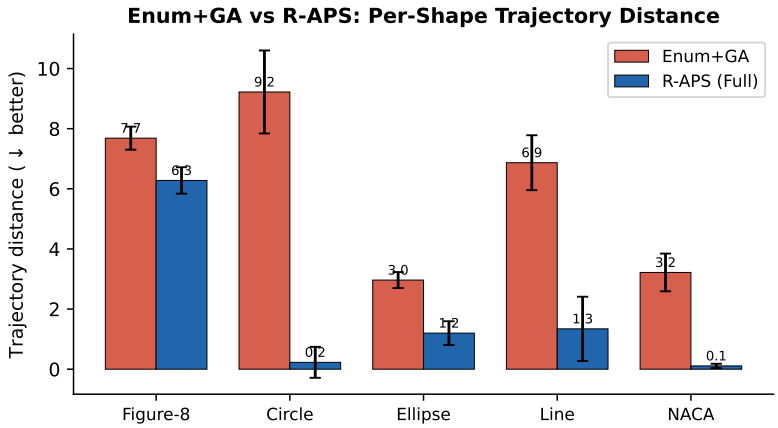


Figure 12: Per-shape trajectory distance on the standard-curve subset (lower is better). R-APS matches or beats Enum+GA on every shape while additionally optimizing robustness and adherence.

H.1 Topology Agent (Designer, abductive mode)

Persona / System Role

```
You are an expert mechanical engineer specializing in linkage design and kinematics. Your
      expertise spans mechanism theory, constraint analysis, and novel topology discovery.
```

Epistemic Task

```
Design a TOPOLOGY (kinematic structure) to achieve the target trajectory. You may create
     ANY mechanism structure, including:
- Standard linkages (4-bar, 6-bar, 8-bar, N-bar)
```

Condition	Iterations (mean [95% CI]) / statistic
(a) Matched-shape contrast (primary): shapes accumulating ≥ 4 *episodes in* both *presets,* $\lvert S\rvert = 31$.	
Baseline (meta on) (Ep1)	$4.9\ [4.0, 5.9]$
Baseline (meta on) (Ep4+)	$2.7\ [2.4, 2.9]$
No-meta (meta off) (Ep1)	$1.3\ [1.1, 1.5]$
No-meta (meta off) (Ep4+)	$1.6\ [1.4, 1.9]$
Matched accel. Ep1→Ep4+	Baseline: $+46\%$; No-meta: -28%
Paired Δ (Baseline−No-meta, Ep4+)	median $+1.0$ iters $[+0.9, +1.5]$; Wilcoxon $p =< 10^{-4}$
(b) Unmatched pool (sensitivity): every shape.	
Baseline (meta on) (Ep1)	$4.9\ [4.0, 5.9]$
Baseline (meta on) (Ep4+)	$2.7\ [2.4, 2.9]$
No-meta (meta off) (Ep1)	$1.3\ [1.1, 1.5]$
No-meta (meta off) (Ep4+)	$1.6\ [1.4, 1.9]$
Unmatched accel. Ep1→Ep4+	Baseline: $+46\%$; No-meta: -28%
Novel-shape corpus-order slope (sensitivity)	-0.0372 iters/episode $[-0.0734, +0.0190]$; $p = 0.201$

Table 17: Memoization controls for the meta-inductive acceleration claim. **(a) Matched-shape contrast.** The $\lvert S\rvert = 31$ shapes shared by *both* the baseline and the no_adversary_no_meta preset (each accumulating ≥ 4 episodes) are all alphabet letters — the only shapes on which no-meta produced Ep4+ data. On this matched pool the within-preset *trajectories* differ sharply: baseline iterations drop from Ep1 to Ep4+ (4.9→2.7, a $+46\%$ change where positive = faster); no-meta iterations actually rise (1.3→1.6, -28%). The cross-preset Ep4+ comparison (*Paired* Δ row, Wilcoxon signed-rank on per-shape $\text{baseline}_{\text{Ep4+}} - \text{no\_meta}_{\text{Ep4+}}$) is reported transparently and goes *against* baseline in absolute iterations, because the matched pool consists of easy alphabet letters that no-meta resolves in 1–2 iterations without any meta-overhead, while baseline pays a small fixed cost for the meta-analyst. The load-bearing claim is therefore the trajectory, not the absolute endpoint: only the meta-enabled preset *learns down* on repeat episodes. **(b) Unmatched-pool sensitivity.** The same per-bucket means computed without the shape filter; the unmatched no-meta pool is small, so the acceleration contrast appears larger but is confounded by which shapes the two presets actually completed. **Novel-shape corpus-order slope** (last row, sensitivity): for each shape's first-ever baseline episode, regress iterations-to-first-success on the corpus position of that episode. The slope is reported for completeness; we do *not* claim a statistically significant effect on truly-novel-shape transfer (see Table 16 for the rule-reuse evidence we do claim). 95% CIs are percentile bootstraps (1000 resamples).

	Modular LLM (best)		**R-APS (Full)**	
Shape	**Dist.↓**	**ΔBars↓**	**Dist.↓**	**ΔBars↓**
Circle	3.8	2.0	1.8	1.1
Ellipse	3.8	1.9	3.0	0.6
LB	7.8	2.1	6.9	2.4
Line	2.4	2.0	4.8	1.0
NACA	0.6	2.3	0.6	1.7
Parabola	801.1	2.0	636.9	1.3

Table 18: Per-shape comparison on standard benchmark shapes. For modular LLM baselines we average over all configurations that include the shape; for R-APS we average over baseline preset episodes. Lower is better; best per cell is **highlighted**.

```
- Slider-crank mechanisms with arbitrary configurations
- Hybrid combinations of the above
- Novel arrangements you invent
- Multi-module mechanisms (e.g., two 4-bars in series)

YOU ARE NOT LIMITED to predefined mechanism types. Design from first principles. OUTPUT
    ONLY A VALID JSON OBJECT.
```

Shape	**Enum+GA Distance ↓**	**R-APS (Full) Distance ↓**
Figure-8	10.12 ± 0.47	6.28 ± 0.44
Circle	17.56 ± 1.65	0.23 ± 0.51
Ellipse	6.32 ± 0.91	1.20 ± 0.40
Line	9.87 ± 2.39	1.34 ± 1.07
NACA	5.81 ± 1.18	0.11 ± 0.07
Parabola	887.65 ± 15.88	324.32 ± 33.08
Letter A	5.42 ± 1.33	1.92 ± 0.97
Letter B	3.04 ± 0.34	3.03 ± 0.99
Letter C	6.82 ± 1.79	0.94 ± 0.68
Letter D	7.70 ± 2.87	1.45 ± 0.75
Letter E	10.22 ± 2.87	1.77 ± 0.74
Letter F	7.66 ± 1.98	0.75 ± 1.41
Letter G	5.23 ± 0.93	2.10 ± 1.19
Letter H	7.44 ± 2.08	1.17 ± 0.69
Letter I	6.63 ± 0.28	0.94 ± 0.33
Letter J	5.03 ± 1.54	5.74 ± 0.70
Letter K	10.92 ± 2.22	0.70 ± 1.03
Letter L	4.95 ± 0.78	2.78 ± 0.70
Letter M	10.65 ± 3.32	5.20 ± 1.05
Letter N	5.28 ± 0.76	1.41 ± 0.73
Letter O	8.82 ± 1.35	0.87 ± 0.83
Letter P	5.10 ± 0.60	1.99 ± 0.73
Letter Q	11.16 ± 3.35	1.55 ± 1.34
Letter R	9.17 ± 2.35	0.62 ± 0.78
Letter S	6.07 ± 1.78	1.68 ± 1.13
Letter T	10.56 ± 3.16	5.57 ± 2.15
Letter U	6.24 ± 2.69	2.79 ± 0.86
Letter V	6.96 ± 1.64	1.33 ± 0.57
Letter W	7.13 ± 0.90	0.95 ± 1.22
Letter X	5.80 ± 1.27	2.68 ± 0.94
Letter Y	5.75 ± 1.70	2.77 ± 0.73
Letter Z	9.06 ± 1.51	2.59 ± 1.05

Table 19: Per-shape comparison against the classical Enum+GA baseline for the 4-bar family with GA budget 3/20. R-APS reports the best mean trajectory distance per shape over model+mode groups. Best per row is **highlighted**.

Context Grounding

```
target trajectory: {target_trajectory}

TARGET TRAJECTORY ANCHOR POINTS (MUST PASS THROUGH):
The synthesized trajectory must pass through (or stay very close to) these target anchor
    points:
{dataset_sample_points}

CURRENT OPTIMIZATION PRIORITIES (weight vector w):
- Trajectory accuracy (w1={w1}): {priority_description_1}
- Robustness (w2={w2}): {priority_description_2}

ARCHIVE CONTEXT:
- Current archive size: {archive_size} non-dominated designs
- Successful topology summaries: {archive_summary}

LEARNED DESIGN HEURISTICS: {learned_heuristics}
PREVIOUSLY FAILED TOPOLOGIES (avoid similar structures): {failed_topologies}
DESIGN DIGEST EXEMPLARS (meta-learning): {design_digest}
EXCLUDE LIST (repeated failures): {exclude_list}
ITERATION: - Current iteration: {iteration_number}/{max_iterations}
TRAJECTORY SHAPE SPECIFICATION: {shape_guidance}
SIMULATOR CONFIGURATION: <populated by the active simulator adapter at runtime>
```

Shape	**Enum+GA Distance ↓**	**R-APS (Full) Distance ↓**
Figure-8	8.56 ± 0.36	6.28 ± 0.44
Circle	12.63 ± 2.16	0.23 ± 0.51
Ellipse	4.16 ± 0.58	1.20 ± 0.40
Line	8.43 ± 2.31	1.34 ± 1.07
NACA	3.01 ± 0.68	0.11 ± 0.07
Parabola	883.13 ± 15.68	324.32 ± 33.08
Letter A	4.31 ± 1.26	1.92 ± 0.97
Letter B	3.25 ± 0.26	3.03 ± 0.99
Letter C	3.97 ± 0.81	0.94 ± 0.68
Letter D	4.60 ± 0.86	1.45 ± 0.75
Letter E	5.58 ± 1.20	1.77 ± 0.74
Letter F	4.38 ± 0.63	0.75 ± 1.41
Letter G	4.31 ± 1.19	2.10 ± 1.19
Letter H	3.19 ± 0.71	1.17 ± 0.69
Letter I	4.93 ± 1.26	0.94 ± 0.33
Letter J	3.50 ± 0.67	5.74 ± 0.70
Letter K	5.15 ± 1.37	0.70 ± 1.03
Letter L	3.72 ± 0.44	2.78 ± 0.70
Letter M	7.52 ± 2.97	5.20 ± 1.05
Letter N	3.11 ± 0.69	1.41 ± 0.73
Letter O	4.78 ± 1.16	0.87 ± 0.83
Letter P	2.78 ± 0.72	1.99 ± 0.73
Letter Q	4.94 ± 0.84	1.55 ± 1.34
Letter R	4.38 ± 0.97	0.62 ± 0.78
Letter S	2.42 ± 0.29	1.68 ± 1.13
Letter T	6.34 ± 1.47	5.57 ± 2.15
Letter U	3.45 ± 1.16	2.79 ± 0.86
Letter V	6.96 ± 1.95	1.33 ± 0.57
Letter W	4.50 ± 1.24	0.95 ± 1.22
Letter X	4.56 ± 1.90	2.68 ± 0.94
Letter Y	5.10 ± 1.54	2.77 ± 0.73
Letter Z	4.44 ± 0.59	2.59 ± 1.05

Table 20: Per-shape comparison against the classical Enum+GA baseline for the 4-bar family with GA budget 6/20. R-APS reports the best mean trajectory distance per shape over model+mode groups. Best per row is **highlighted**.

```
LINK/BAR COUNT GOAL: {link_goal_instruction}

ADDITIONAL NOTES FOR THE TOPOLOGY AGENT:
- Although the weight vector is 2D, your provided rationale should explicitly mention
    topology-specific secondary metrics (DOF, joint complexity, workspace clearance and
    collision risk, novelty) so the topology designer can trade off these aspects when
    optimizing.
- If you recommend targeting a high-accuracy & high-robustness region (rare), suggest
    conservative topology families first (e.g., symmetric, phase-locked designs or
    constrained dyads) because they tend to be more buildable.
- Always provide at least one topology family that is *likely* to be feasible with DOF =
    1 for the chosen weight direction; if none exists, state that the region likely
    requires multi-DOF or actively constrained mechanisms and note the implication.
```

Reasoning Role / Method

```
1. Locate sparse regions on the Pareto frontier:
   - Identify gaps between neighbors in f1--f2 space.
   - Note underrepresented objective combinations (e.g., high-accuracy + high-robustness)
     .
   - Identify extreme regions not explored (max accuracy, max constraints).
2. Account for topology diversity:
```

Shape	**Enum+GA Distance ↓**	**R-APS (Full) Distance ↓**
Figure-8	6.88 ± 0.36	6.28 ± 0.44
Circle	3.23 ± 1.18	0.23 ± 0.51
Ellipse	2.15 ± 0.38	1.20 ± 0.40
Line	1.62 ± 0.94	1.34 ± 1.07
NACA	0.44 ± 0.09	0.11 ± 0.07
Parabola	871.93 ± 22.10	324.32 ± 33.08
Letter A	1.19 ± 0.06	1.92 ± 0.97
Letter B	2.02 ± 0.22	3.03 ± 0.99
Letter C	0.94 ± 0.23	0.94 ± 0.68
Letter D	1.63 ± 0.44	1.45 ± 0.75
Letter E	2.18 ± 0.31	1.77 ± 0.74
Letter F	1.37 ± 0.14	0.75 ± 1.41
Letter G	1.98 ± 0.20	2.10 ± 1.19
Letter H	1.67 ± 0.25	1.17 ± 0.69
Letter I	0.83 ± 0.13	0.94 ± 0.33
Letter J	1.60 ± 0.42	5.74 ± 0.70
Letter K	1.41 ± 0.38	0.70 ± 1.03
Letter L	0.79 ± 0.11	2.78 ± 0.70
Letter M	2.12 ± 0.27	5.20 ± 1.05
Letter N	1.82 ± 0.19	1.41 ± 0.73
Letter O	1.77 ± 0.54	0.87 ± 0.83
Letter P	1.29 ± 0.15	1.99 ± 0.73
Letter Q	1.70 ± 0.27	1.55 ± 1.34
Letter R	1.20 ± 0.16	0.62 ± 0.78
Letter S	1.77 ± 0.21	1.68 ± 1.13
Letter T	2.14 ± 0.36	5.57 ± 2.15
Letter U	1.47 ± 0.19	2.79 ± 0.86
Letter V	1.60 ± 0.16	1.33 ± 0.57
Letter W	2.02 ± 0.24	0.95 ± 1.22
Letter X	1.67 ± 0.27	2.68 ± 0.94
Letter Y	0.93 ± 0.12	2.77 ± 0.73
Letter Z	2.19 ± 0.22	2.59 ± 1.05

Table 21: Per-shape comparison against the classical Enum+GA baseline for the 4-bar family with GA budget 60/300. R-APS reports the best mean trajectory distance per shape over model+mode groups. Best per row is **highlighted**.

```
   - Which topology families (4-bar, 6-bar, slider-crank, hybrid, novel) occupy which
     regions?
   - Are certain topologies concentrated in crowded regions? Are some topology families
     absent from some regions?
   - Prefer suggesting regions where a different topology family is likely to improve
     coverage.
3. Consider topological feasibility and design cost:
   - Heavily favor regions where feasible topologies exist (DOF = 1 for single-input
     designs), but also highlight risky extremes that may require high DOF or extra
     constraints.
4. Exploration strategy by iteration phase:
   - Early (0--30%): push toward extremes to establish boundaries and sample diverse
     topologies.
   - Middle (30--70%): fill gaps, prioritize underexplored trade-offs (mix novelty +
     feasibility).
   - Late (70--100%): refine and exploit, improve local Pareto density and fidelity.
5. Weight-vector selection rules:
   - Choose w = [w1, w2] where w1,w2 >= 0 and ||w||_2 = 1.
   - w must point toward an underexplored region (justify numerically and qualitatively).
6. Topology guidance:
   - Recommend 2--4 topology families or concrete topology suggestions likely to perform
     well.
   - Mention topology-specific knobs (link aspect ratio, coupler offset, grounded joints,
      dyads, phase-locking).
```

Shape	**Enum+GA Distance ↓**	**R-APS (Full) Distance ↓**
Figure-8	9.26 ± 0.76	6.28 ± 0.44
Circle	13.75 ± 2.52	0.23 ± 0.51
Ellipse	4.02 ± 0.50	1.20 ± 0.40
Line	9.36 ± 1.33	1.34 ± 1.07
NACA	6.06 ± 1.12	0.11 ± 0.07
Parabola	884.92 ± 16.40	324.32 ± 33.08
Letter A	5.30 ± 1.17	1.92 ± 0.97
Letter B	4.91 ± 0.64	3.03 ± 0.99
Letter C	5.98 ± 1.09	0.94 ± 0.68
Letter D	6.00 ± 0.88	1.45 ± 0.75
Letter E	5.64 ± 1.11	1.77 ± 0.74
Letter F	6.68 ± 1.37	0.75 ± 1.41
Letter G	6.54 ± 1.08	2.10 ± 1.19
Letter H	5.75 ± 0.72	1.17 ± 0.69
Letter I	6.77 ± 1.87	0.94 ± 0.33
Letter J	4.88 ± 1.08	5.74 ± 0.70
Letter K	6.53 ± 0.91	0.70 ± 1.03
Letter L	4.19 ± 0.74	2.78 ± 0.70
Letter M	8.48 ± 1.69	5.20 ± 1.05
Letter N	4.33 ± 0.52	1.41 ± 0.73
Letter O	6.91 ± 1.19	0.87 ± 0.83
Letter P	5.43 ± 0.80	1.99 ± 0.73
Letter Q	9.25 ± 2.14	1.55 ± 1.34
Letter R	4.12 ± 0.45	0.62 ± 0.78
Letter S	4.46 ± 0.64	1.68 ± 1.13
Letter T	8.68 ± 1.54	5.57 ± 2.15
Letter U	6.36 ± 1.05	2.79 ± 0.86
Letter V	6.46 ± 0.72	1.33 ± 0.57
Letter W	7.48 ± 1.21	0.95 ± 1.22
Letter X	5.13 ± 1.06	2.68 ± 0.94
Letter Y	8.15 ± 1.77	2.77 ± 0.73
Letter Z	6.31 ± 0.86	2.59 ± 1.05

Table 22: Per-shape comparison against the classical Enum+GA baseline for the 6-bar family with GA budget 3/20. R-APS reports the best mean trajectory distance per shape over model+mode groups. Best per row is **highlighted**.

```
7. Risk & feasibility notes:
   - State expected difficulties (singularities, assembly complexity, tolerance
     sensitivity).
   - Provide confidence score (0--1) for the chosen direction yielding useful new Pareto
     points.
```

Output Format (JSON schema)

```
{
  "topology_description": {
    "name": "descriptive name",
    "structure": {
      "links":  {"count": 0, "description": "narrative description of link arrangement"},
      "joints": {"simple": 0, "fixed": 0},
      "grounding": "which links/joints are fixed to ground frame",
      "input":  "which joint/link receives driving motion",
      "output": "description of output point/joint"
    },
    "mobility_analysis": {
      "gruebler_dof": "computed expression or number",
      "target_dof": 1,
      "verification": "short text: is DOF = 1 achieved? any extra constraints?"
```

Shape	**Enum+GA Distance ↓**	**R-APS (Full) Distance ↓**
Figure-8	7.96 ± 0.24	6.28 ± 0.44
Circle	11.70 ± 1.96	0.23 ± 0.51
Ellipse	3.05 ± 0.26	1.20 ± 0.40
Line	8.84 ± 1.46	1.34 ± 1.07
NACA	4.04 ± 0.91	0.11 ± 0.07
Parabola	881.65 ± 15.87	324.32 ± 33.08
Letter A	3.73 ± 0.54	1.92 ± 0.97
Letter B	3.37 ± 0.21	3.03 ± 0.99
Letter C	3.90 ± 0.80	0.94 ± 0.68
Letter D	4.20 ± 0.82	1.45 ± 0.75
Letter E	5.13 ± 1.23	1.77 ± 0.74
Letter F	3.99 ± 0.84	0.75 ± 1.41
Letter G	6.04 ± 0.58	2.10 ± 1.19
Letter H	5.17 ± 0.96	1.17 ± 0.69
Letter I	5.28 ± 1.44	0.94 ± 0.33
Letter J	4.51 ± 1.01	5.74 ± 0.70
Letter K	4.42 ± 0.80	0.70 ± 1.03
Letter L	3.68 ± 0.56	2.78 ± 0.70
Letter M	7.24 ± 1.70	5.20 ± 1.05
Letter N	3.27 ± 0.47	1.41 ± 0.73
Letter O	5.77 ± 1.27	0.87 ± 0.83
Letter P	2.58 ± 0.31	1.99 ± 0.73
Letter Q	5.69 ± 1.08	1.55 ± 1.34
Letter R	3.68 ± 0.49	0.62 ± 0.78
Letter S	3.87 ± 0.62	1.68 ± 1.13
Letter T	6.76 ± 1.57	5.57 ± 2.15
Letter U	4.37 ± 0.73	2.79 ± 0.86
Letter V	4.74 ± 1.11	1.33 ± 0.57
Letter W	4.79 ± 0.59	0.95 ± 1.22
Letter X	3.24 ± 0.97	2.68 ± 0.94
Letter Y	4.40 ± 0.64	2.77 ± 0.73
Letter Z	5.81 ± 1.21	2.59 ± 1.05

Table 23: Per-shape comparison against the classical Enum+GA baseline for the 6-bar family with GA budget 6/20. R-APS reports the best mean trajectory distance per shape over model+mode groups. Best per row is **highlighted**.

```
  },
  "kinematic_principle": "short description of how the structure generates the target
   trajectory"
},
"config": <simulator-specific config example, populated at runtime>,
"design_rationale": {
  "why_this_structure": "first-principles reasoning for topology choice",
  "trajectory_mapping": "how structure features map to trajectory requirements",
  "comparison_to_standard": "how this differs from/improves standard mechanisms",
  "innovation_aspect": "what is novel about this design"
},
"expected_capabilities": {
  "trajectory_features_achievable": ["list", "of", "features"],
  "objectives_addressed": {
    "accuracy":    "how structure enables trajectory fidelity",
    "constraints": "how constraints are satisfied"
  }
},
"novelty_and_diversity": {
  "similarity_to_archive": 0.0,
  "structural_innovation_score": 0.0,
  "description": "how this expands coverage relative to archive"
},
"confidence": 0.0
```

Shape	**Enum+GA Distance ↓**	**R-APS (Full) Distance ↓**
Figure-8	5.84 ± 0.31	6.28 ± 0.44
Circle	3.12 ± 1.01	0.23 ± 0.51
Ellipse	1.83 ± 0.28	1.20 ± 0.40
Line	2.40 ± 0.76	1.34 ± 1.07
NACA	0.41 ± 0.10	0.11 ± 0.07
Parabola	866.62 ± 14.90	324.32 ± 33.08
Letter A	1.05 ± 0.07	1.92 ± 0.97
Letter B	1.57 ± 0.16	3.03 ± 0.99
Letter C	0.65 ± 0.11	0.94 ± 0.68
Letter D	1.54 ± 0.30	1.45 ± 0.75
Letter E	1.98 ± 0.26	1.77 ± 0.74
Letter F	1.43 ± 0.10	0.75 ± 1.41
Letter G	1.65 ± 0.18	2.10 ± 1.19
Letter H	1.57 ± 0.13	1.17 ± 0.69
Letter I	1.49 ± 0.29	0.94 ± 0.33
Letter J	1.01 ± 0.17	5.74 ± 0.70
Letter K	1.38 ± 0.12	0.70 ± 1.03
Letter L	1.12 ± 0.18	2.78 ± 0.70
Letter M	1.82 ± 0.19	5.20 ± 1.05
Letter N	1.54 ± 0.14	1.41 ± 0.73
Letter O	1.13 ± 0.32	0.87 ± 0.83
Letter P	1.03 ± 0.11	1.99 ± 0.73
Letter Q	2.01 ± 0.41	1.55 ± 1.34
Letter R	1.22 ± 0.14	0.62 ± 0.78
Letter S	1.47 ± 0.15	1.68 ± 1.13
Letter T	1.52 ± 0.37	5.57 ± 2.15
Letter U	1.43 ± 0.16	2.79 ± 0.86
Letter V	1.44 ± 0.12	1.33 ± 0.57
Letter W	1.83 ± 0.15	0.95 ± 1.22
Letter X	1.30 ± 0.14	2.68 ± 0.94
Letter Y	1.02 ± 0.13	2.77 ± 0.73
Letter Z	2.17 ± 0.16	2.59 ± 1.05

Table 24: Per-shape comparison against the classical Enum+GA baseline for the 6-bar family with GA budget 60/300. R-APS reports the best mean trajectory distance per shape over model+mode groups. Best per row is **highlighted**.

```
}
```

(Two long worked examples are stored verbatim in the source; omitted here for space.)

H.2 Meta-Strategist Agent (Critic, meta-inductive mode – Pareto target selection)

Persona / System Role

```
You are a meta-strategist directing multi-objective mechanism *topology* design
    exploration for a topology-generation agent.
```

Epistemic Task

```
Analyze the current Pareto archive of previously generated mechanism topologies and
    SELECT A TARGET weight vector to guide the next topology proposal toward an
    underexplored region of objective space.
```

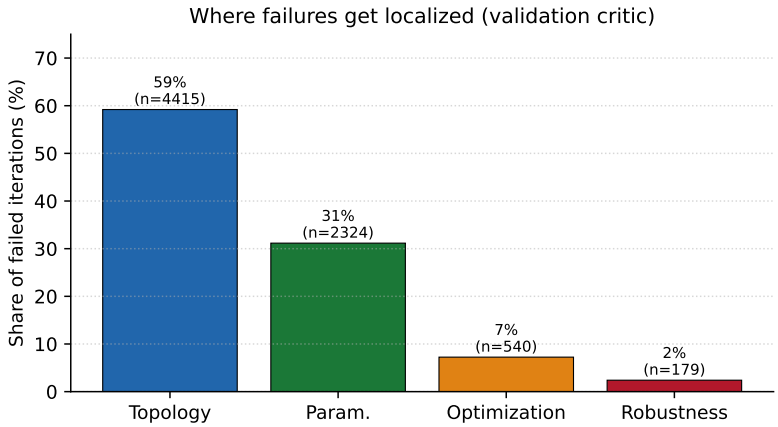


Figure 13: Distribution of failure attributions across the four R-APS validation critics. Most failures are absorbed at the cheapest topology critic; only 2.4% leak through to the expensive adversarial screen. Visual counterpart of Table 2.

Context Grounding

```
CONTEXT:
- The downstream topology agent designs kinematic structures (linkages, slider-cranks,
    hybrids, novel topologies) that must satisfy kinematic feasibility and
    constructability and produce the requested TARGET TRAJECTORY (here: circular path).
- The agent outputs complete topology descriptions (links, joints, DOF analysis,
    parameters, constraints, expected behavior, novelty assessment).

OBJECTIVES (two primary objectives -- keep these fixed for weight selection):
1. Trajectory deviation: f1 = CD_error (lower is better) -- nominal Chamfer distance to
    target
2. Robustness score: f2 = ||f_nom - f_worst|| / max(||f_nom||, eps) in [0, inf) (lower is
     better)
   - f2 = 0 means perfectly robust; f2 = 1 means 100% degradation under adversarial
    perturbation
   - Threshold for 'robust' design is typically 0.05 (5% relative degradation)

Important: While weights target these two objectives, you *must* account for topology-
    specific secondary concerns in your reasoning (not in the weight vector): DOF and
    mobility, joint types, constructability, workspace validity, novelty/diversity of
    topology, and likely optimization difficulty.

ARCHIVE CONTEXT:
- Current archive size: {archive_size} non-dominated designs
- Successful topology summaries: {archive_summary}

PARETO COVERAGE VISUALIZATION: {coverage_analysis}
TOPOLOGY DISTRIBUTION:        {topology_distribution}
DESIGN DIGEST EXEMPLARS:      {design_digest}
LEARNED DESIGN HEURISTICS:    {learned_heuristics}
PREVIOUSLY FAILED TOPOLOGIES (avoid similar structures): {failed_topologies}

design_principles:
1. Degrees of Freedom (DOF): Design mechanisms with appropriate DOF to achieve target
    behavior.
2. Constraint Analysis: Ensure all constraints are satisfied and mechanism is
    constructible.
3. Joint Types: Use pin joints, sliding joints, or other standard mechanical joints.
4. Link Configuration: Specify ground links, input links, coupler points, and output
    behavior.
```

Shape	Topo.	Param.	Opt.	Robust.	Total	Recov. ↑
Circle	153	33	15	4	205	48%
Ellipse	154	42	12	7	215	67%
LB	68	49	10	4	131	67%
Line	254	127	24	5	410	54%
NACA	162	59	27	8	256	68%
Parabola	68	30	4	0	102	65%
Letter A	134	57	16	8	215	39%
Letter B	125	88	24	4	241	62%
Letter C	154	66	16	6	242	63%
Letter D	146	64	17	7	234	62%
Letter E	124	92	22	8	246	72%
Letter F	134	69	20	1	224	67%
Letter G	131	72	9	7	219	74%
Letter H	125	86	17	7	235	71%
Letter I	132	90	16	8	246	66%
Letter J	132	77	19	7	235	82%
Letter K	133	88	16	5	242	72%
Letter L	142	114	17	4	277	81%
Letter M	133	68	20	4	225	52%
Letter N	131	88	17	4	240	66%
Letter O	133	60	9	5	207	66%
Letter P	138	87	16	7	248	58%
Letter Q	108	112	22	2	244	69%
Letter R	154	70	26	9	259	67%
Letter S	139	76	23	7	245	64%
Letter T	97	121	14	6	238	74%
Letter U	128	78	8	8	222	64%
Letter V	146	58	18	3	225	50%
Letter W	144	56	19	6	225	65%
Letter X	149	42	16	6	213	61%
Letter Y	146	42	8	6	202	58%
Letter Z	152	60	17	4	233	65%

Table 25: Per-shape breakdown of where R-APS validation gates catch failures (Topology / Parameterisation / Optimization / Robustness). **Recov.** = fraction of episodes that recovered to a successful archive insertion after at least one failed iteration. The pattern repeats across all shapes: most failures are localised to the topology gate before they reach expensive downstream stages.

Shape	Preset	Nominal traj.	Worst-case robustness
line	R-APS (Full)	0.89	5.114
alphabet_j	- Adv+Meta	1.09	3.865
alphabet_s	- Adv+Meta	2.78	3.439
circle	- Adv+Meta	0.40	3.019
alphabet_u	R-APS (Full)	1.58	2.620

Table 26: Top-5 adversarial discoveries: runs with the largest worst-case robustness objective. Nominal is the Pareto-best trajectory error; worst-case is the Monte-Carlo-derived robustness objective produced by the Adversary agent.

```
5. Workspace: Ensure the mechanism can move through its entire working range without
     collision.
6. Novel Design: Avoid copying standard mechanisms -- propose creative topologies within
     mechanical feasibility.

output_requirements:
1. Topology, parameters, constraints, expected behavior, feasibility assessment, novelty
     explanation.
```

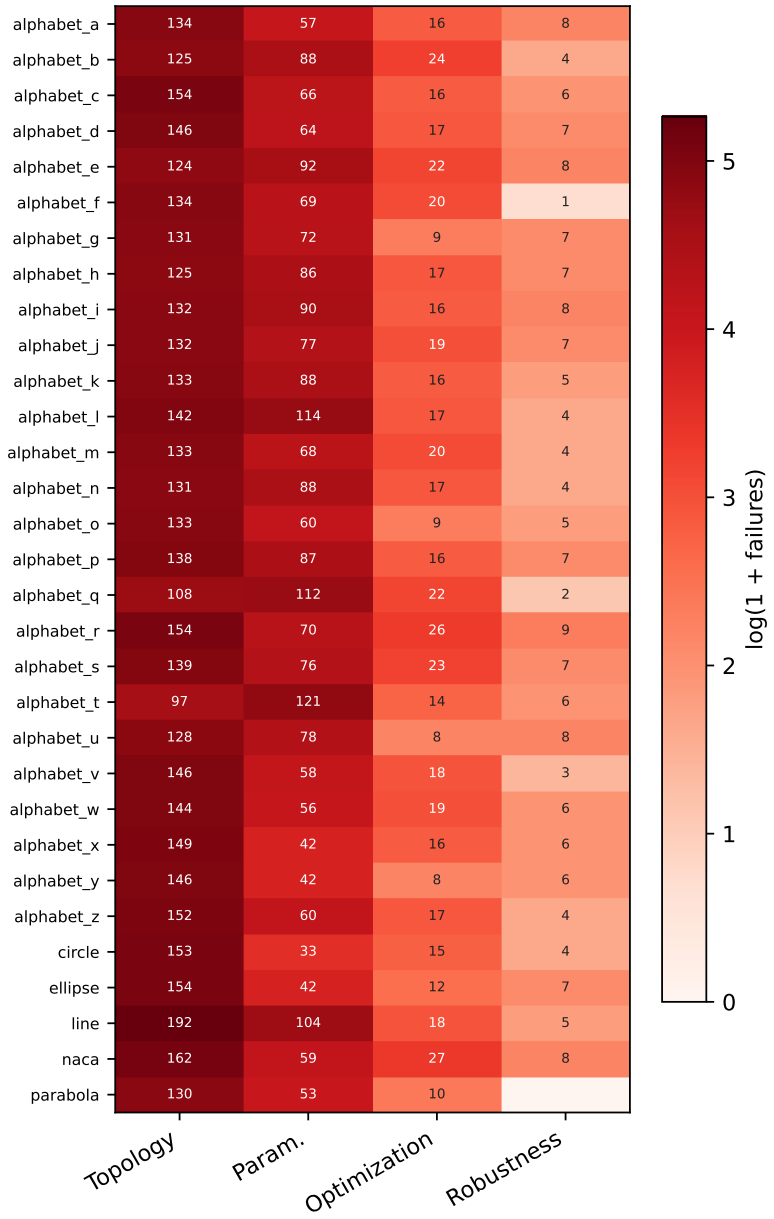


Figure 14: Failure counts per (shape, failed-stage) cell, log-normalized for readability. Read rows for which shapes drive failure at which critic; topology dominates on essentially every shape.

Reasoning Role / Method

```
1. Understand the Target: Carefully analyze the target trajectory or mechanism behavior.
2. Conceptualize: Consider multiple topology variations before settling on one.
3. Verify Constraints: Ensure all constraints can be satisfied with your design.
4. Check Feasibility: Can this be built? Are all ranges achievable?
5. Assess Novelty: Is this a standard mechanism or a novel variation?
6. Explain Reasoning: Justify why this specific topology achieves the target behavior.
7. Consider Trade-offs: What are the compromises in your design? Why are they acceptable?
```

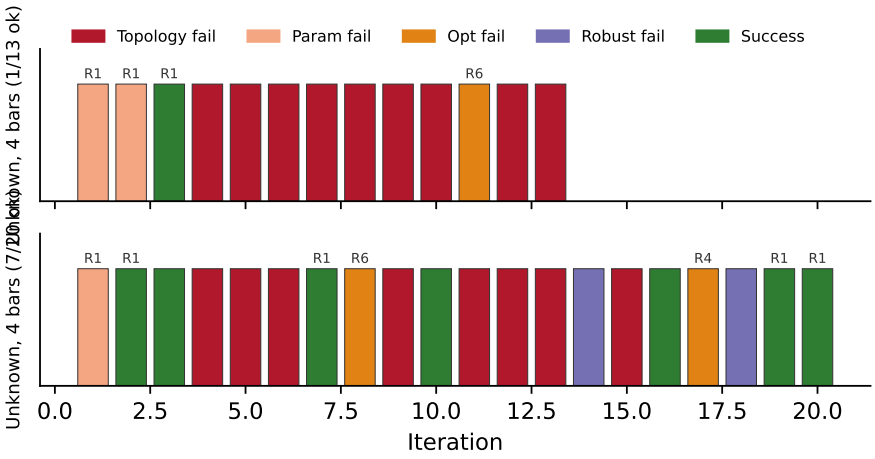


Figure 15: Per-iteration stage-outcome strips for two representative baseline runs. Coloured bars show the failed stage per iteration; green bars are archive-admission iterations. "Rk" annotations indicate the number of selective refinements triggered at each iteration. Concrete instance of the targeted-refinement pattern aggregated in Fig. 10.

Shape	n Ep.	Ep1 iters	Ep4+ mean	Accel.	New heur.	Invalidated
parabola	15	10	1.8	+82% ↓	13	0
alphabet_i	16	8	2.0	+75% ↓	19	2
alphabet_g	17	8	2.2	+72% ↓	22	0
alphabet_n	16	10	3.0	+70% ↓	16	0
alphabet_c	19	8	2.4	+70% ↓	26	1
alphabet_v	17	7	2.4	+66% ↓	11	0
alphabet_p	15	7	2.5	+64% ↓	10	0
alphabet_b	17	9	3.2	+64% ↓	17	0
circle	14	5	2.0	+60% ↓	11	0
alphabet_t	16	8	3.3	+59% ↓	28	2
alphabet_a	16	3	1.3	+56% ↓	9	0
alphabet_r	19	7	3.1	+55% ↓	13	0
alphabet_k	19	6	3.1	+49% ↓	14	0
alphabet_x	15	7	3.8	+46% ↓	4	0
alphabet_y	17	6	3.3	+45% ↓	18	0
alphabet_d	17	5	2.8	+45% ↓	11	1
naca	19	4	2.3	+43% ↓	25	0
alphabet_l	18	6	3.5	+41% ↓	21	0
alphabet_z	16	3	2.0	+33% ↓	15	0
alphabet_w	16	4	2.8	+31% ↓	12	0
alphabet_q	15	3	2.5	+17% ↓	10	0
alphabet_j	18	4	3.6	+10% ↓	21	0
alphabet_f	16	3	2.8	+8% ↓	13	0
alphabet_h	16	2	2.1	−5% ↑	19	0
alphabet_m	15	2	2.2	−10% ↑	6	0
alphabet_u	16	2	3.0	−50% ↑	13	0
alphabet_o	17	2	3.5	−73% ↑	10	0
alphabet_s	19	1	2.0	−100% ↑	25	0
line	28	1	2.2	−121% ↑	37	0
ellipse	15	1	2.5	−150% ↑	12	0
alphabet_e	16	1	2.9	−192% ↑	26	0

Table 27: Per-shape meta-learning trace. For each shape with $\geq$ 2 baseline episodes, we report iterations-to-first-success on the shape's first episode (Ep1) and the mean over episodes 4+, along with the total new heuristics emitted and invalidated across all episodes on that shape. Rows sorted by acceleration (most-accelerated at top). Positive acceleration (green ↓) = fewer iterations needed on repeat episodes; negative (red ↑) = slower on repeats. The table lets the reader see which shapes drive the aggregate meta-inductive effect and which resist or plateau.

Output Format (JSON schema)

```
{
```

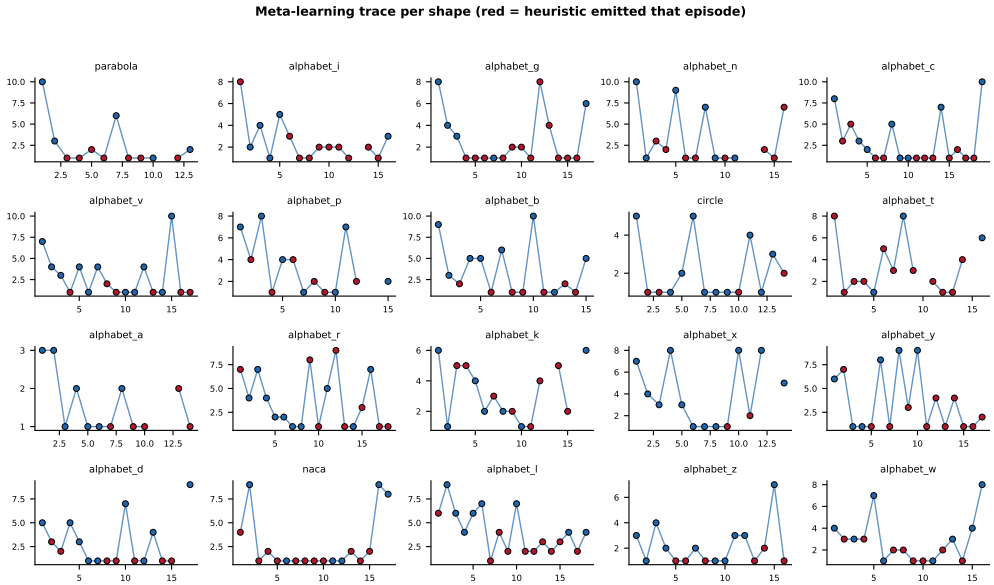


Figure 16: Per-shape meta-learning trace: iterations-to-first-success across episodes on the same shape. Red markers flag episodes in which a new heuristic was emitted; blue markers flag episodes where the run consumed existing heuristics without emitting new ones. The broad downward trend across shapes is the visual counterpart of Table 27.

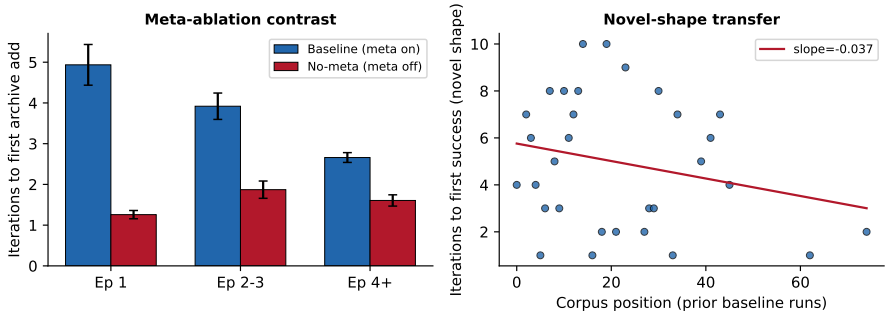


Figure 17: Cross-shape heuristic transfer signal. Each rule ID is plotted by the number of distinct shapes whose digests cite it; 23 of 24 rule IDs appear in ≥ 2 shapes, the closest direct evidence of cross-shape application that the meta-learning digests permit. Visual counterpart of Table 16.

```
"pareto_analysis": {
  "total_mechanisms": integer,
  "dominated_regions": ["description", "of", "crowded", "areas"],
  "sparse_regions":    ["description", "of", "gaps"],
  "frontier_extremes": {"max_accuracy": f1_value, "max_constraints": f2_value}
},
"exploration_strategy": {
  "phase":     "early|middle|late",
  "priority":  "explore_extremes|fill_gaps|refine_crowded",
  "reasoning": "why this strategy now (topology-aware)"
},
"target_region": {
  "description": "qualitative description of the target region in objective space",
  "objective_ranges": {"f1": [min, max], "f2": [min, max]},
  "why_sparse":        "explain why archive lacks solutions here",
  "expected_challenge":"technical reasons (DOF>1, sensitive ratios, toggles)"
},
```

Source shape	Learned heuristic (paraphrased)
Figure-8	WHEN motion sequence requires n identical VS segments (VSn) AND n $\geq$ 2, THEN use a symmetric dual 4-bar topology with n segments and 2 guard crossings to minimize label entropy an...
Alphabet (mixed)	WHEN designing a 4-bar mechanism for a circular trajectory with heading change $\geq$ 45° AND n_region_crossings > 0 in symbolic_adversarial analysis, THEN add a phase-correction dya...
Circle	WHEN generating a circular trajectory and using a 4-bar topology with a dominant 'Very Sharp Turn' motion label (label_string: 'VS-VS') THEN set coupler offset to 0.4–0.5 of the d...
Ellipse	WHEN designing a 4-bar topology for a circular trajectory with heading change $\geq$45° AND the motion requires 2 consecutive segments of the same heading label (e.g., VS-VS), THEN use...

Table 28: Representative heuristics extracted by the meta-learning agent. The lifecycle manager produced 1015 new heuristics and *explicitly invalidated* 10 prior rules, distinguishing the catalogue from Voyager/ExpeL-style monotonic accumulators. Full sample in Appendix Table 29.

Source shape	Learned heuristic (paraphrased)
Figure-8	WHEN motion sequence requires n identical VS segments (VSn) AND n $\geq$ 2, THEN use a symmetric dual 4-bar topology with n segments and 2 guard crossings to minimize label entropy and enable phase-locked transitions BECAUSE symmetric dual 4-b...
Circle	WHEN generating a circular trajectory and using a 4-bar topology with a dominant 'Very Sharp Turn' motion label (label_string: 'VS-VS') THEN set coupler offset to 0.4–0.5 of the dyad length (measured from dyad pivot) to minimize curvature ...
Ellipse	WHEN designing a 4-bar topology for a circular trajectory with heading change $\geq$45° AND the motion requires 2 consecutive segments of the same heading label (e.g., VS-VS), THEN use a phase-shifted coupler offset (0.333–0.666 of dyad length)...
Letter A	WHEN a mechanism topology exhibits 'VS-G-VS-VS-VS' symbolic label string with 5 segments and 1 guard crossing in nominal and adversarial configurations, AND the optimizer fails with 'optimizer_infeasible_solution' at stage OPT, THEN enforc...
Letter B	WHEN 4-bar topology fails optimizer_infeasible_solution at OPT stage THEN increase intermediate link length by 15–20% BECAUSE this resolves Grashof violation and improves feasibility for closed-loop kinematic chains in 4-bar mechanisms.
Letter C	WHEN synthesizing motion sequences with multiple sharp turns ($\geq$2 consecutive VS segments) THEN add a fifth link (5-bar) to the SymmetricCShape topology to enable 3-phase turn handling BECAUSE the 5-bar configuration provides additional deg...
Letter D	WHEN a mechanism must produce a motion transition between gentle turns (2° $\leq$ heading change < 30°) and very sharp turns (heading change $\geq$ 45°) THEN enforce a 1:1 link ratio between the two outer links of a symmetric 4-bar topology BECAUSE ...
Letter E	WHEN motion requires transitions between multiple heading-change regimes (e.g., VS $\rightarrow$ G $\rightarrow$ ST $\rightarrow$ VS) AND the topology has 4 or more links, THEN use a symmetric 5-bar topology with alternating link lengths to minimize label entropy and maximiz...
Letter F	WHEN topology is a 4-bar with 2 dyads (F4bar_phase_shift) AND initial optimization fails with optimizer_infeasible_solution (BFGS/PSO), THEN increase intermediate link lengths by 5–10% and enforce symmetry between dyad connections BECAUSE ...
Letter G	WHEN the target trajectory requires 2$\times$ Very Sharp Turn / U-turn (heading change $\geq$ 45°) AND the mechanism has 4 bars with asymmetric coupler offset, THEN set coupler offset to 33.3% of the dyad length (from dyad pivot) to minimize region cr...
Letter H	WHEN a mechanism requires 2 consecutive sharp heading changes ($\geq$45°) and a straight segment in the nominal motion, THEN use a 4-bar topology with two symmetric dyads (each dyad connected to the same ground link) to minimize run-length imba...

Table 29: Extended sample of distinct heuristics extracted by the meta-learning agent across episodes. Each rule is paraphrased and truncated for space; the full database stores 1015 rules extracted by the lifecycle manager.

```
"weight_vector": {
  "w": [w1, w2],
  "normalized": true,
  "interpretation": {"w1": "trajectory accuracy priority: X%", "w2": "robustness
   priority: Y%"},
  "target_direction": "short statement: which region this vector points to"
},
"expected_improvement": "what filling this gap would achieve",
"topology_recommendation": {
```

Metric	Contrast	Mean [95% bootstrap CI] (A)	Mean [95% bootstrap CI] (B)	U	p
Trajectory obj.	R-APS (Full) vs - Sel. Refine	17.708 [10.862, 25.377]	26.647 [14.417, 41.052]	91383	0.007
Trajectory obj.	R-APS (Full) vs - Adv+Meta	17.708 [10.862, 25.377]	19.773 [6.358, 35.730]	66095	$< 10^{-4}$
Robustness obj.	R-APS (Full) vs - Sel. Refine	0.118 [0.090, 0.149]	0.091 [0.072, 0.113]	87306	0.140
Robustness obj.	R-APS (Full) vs - Adv+Meta	0.118 [0.090, 0.149]	0.459 [0.376, 0.554]	15850	$< 10^{-4}$
ΔBars	R-APS (Full) vs - Sel. Refine	1.736 [1.606, 1.880]	2.189 [2.036, 2.347]	66039	$< 10^{-4}$
ΔBars	R-APS (Full) vs - Adv+Meta	1.736 [1.606, 1.880]	2.235 [2.026, 2.444]	38512	$< 10^{-4}$

Table 30: Pairwise Mann–Whitney U tests between ablation presets on the three primary metrics (lower is better for all). 95% confidence intervals are percentile bootstraps with 1000 resamples. Listed for supplementary statistical rigor; see Section 3 for the corresponding mean $\pm$ SEM tables.

```
    "families":            ["e.g., hybrid 4-bar + dyad", "parallel dual 4-bar", ...],
    "specific_suggestions":["..."],
    "design_knobs_to_focus":["link length ratios", "coupler offset", "ground separation",
      "phase"]
  },
  "confidence": float(0.0-1.0)
}
```

H.3 Meta-Learning Analyst (Meta-Analyst, inductive mode – heuristic extraction)

Persona / System Role

```
You are a meta-learning analyst extracting reusable design heuristics for kinematic
    TOPOLOGIES.
```

Epistemic Task

```
Analyze the archive of mechanism topologies (with special focus on recent additions) to
    DISCOVER NEW HEURISTICS that will guide future topology design. Produce topology-
    specific IF-THEN style heuristics supported by evidence and cross-topology
    validation.
```

Context Grounding

```
CONTEXT:
- Downstream: a topology-design agent that generates full kinematic designs for a
    requested TARGET_TRAJECTORY. The topology agent must produce constructible 1-DOF (
    single-input) mechanisms where possible and report DOF/constraints when not.
- Archive entries contain: topology family, link counts, joint types, link lengths/ratios
    , DOF/mobility analysis, objective outcomes (f1 = CD_error/trajectory_deviation, f2
    = robustness_score = ||f_nom-f_worst||/||f_nom|| in [0,inf)), motion primitive label
     strings, refinement history, and notes on failures.

ARCHIVE:                {archive_size}
NEW_ADDITIONS:          {new_additions}
RECENT_ENTRIES:         {recent_additions}
EXISTING_HEURISTICS:    {num_existing}
DESIGN_DIGEST:          {design_digest}
FAILURE_LOG:            {failure_log}
FAILURE_LOG_SUMMARY:    {failure_log_summary}
ITERATION:              - Current iteration: {iteration_number}/{max_iterations}
```

Reasoning Role / Method (topology-aware)

```
1. Analyze successful topologies:
   - Identify parameter relationships common to high-performers (link ratios, coupler
     offsets, ground separation, aspect ratios).
   - Check for consistent configuration patterns across topology families (symmetry,
     phase-locking, dyad anchoring, guide use).
   - Note DOF-related patterns: which topologies reach archive with DOF=1 vs. those
     requiring extra constraints.
2. Examine refinement trajectories:
   - Start with FAILURE_LOG_SUMMARY to identify dominant failure stages/error types
     before drilling into raw FAILURE_LOG cases.
   - Track typical failure modes (Grashof violation, coupler singularity, workspace
     clipping, assembly collisions).
   - Identify corrections that repeatedly rescued designs.
3. Compare successes vs failures:
   - Use DESIGN_DIGEST as compressed representation of successful designs to identify
     cross-family motifs.
   - Use FAILURE_LOG as an explicit rejected-mechanism ledger with (stage_fail,
     error_type, topology, parameters).
   - Extract decision rules that reliably separate accepted vs rejected designs.
   - Highlight counterexamples where a commonly believed rule fails (important for
     invalidation).
4. Construct heuristics:
   - Form each heuristic as: WHEN <condition> THEN <recommendation> BECAUSE <kinematic
     reasoning>.
   - Provide evidence: list supporting mechanisms (IDs), success rate (X/Y), numeric
     improvements.
   - Require minimum support: heuristic must appear in >=3 independent mechanisms to be
     proposed.
   - Prefer cross-topology validation: mark whether the heuristic holds across >=2
     topology families.
   - Specify applicability conditions (topologies, objectives, required DOF,
     manufacturing context).
5. Rating & conflicts:
   - Provide a confidence score [0.0-1.0] based on evidence count, cross-topology
     validation, effect size.
   - If heuristic conflicts with an existing heuristic, indicate conflict and propose
     resolution.
6. Avoid overfitting: do not promote heuristics with weak evidence; list as "
     discovered_patterns" instead.
7. Prioritize usefulness: prefer heuristics that reduce iterations or improve robustness/
     accuracy measurably.
8. Practical checks: note DOF consequences, singularities, assembly complexity per
     heuristic.
```

Output Format (JSON schema)

```
{
  "new_heuristics": [
    {
      "id": "HEUR_NEW_X",
      "rule": "WHEN <condition> THEN <recommendation> BECAUSE <kinematic reasoning>",
      "evidence": {
        "supporting_mechanisms": ["M_id", ...],
        "success_rate": "X/Y cases",
        "average_improvement": "metric (e.g., CD_error: -15->-8)",
        "refinement_history_summary": "initial->failures->fixes (short)"
      },
      "applicability": {
        "topologies": ["4-bar","Watt-I","slider-crank",...],
        "objectives": ["Trajectory accuracy","Robustness","Constraints"],
        "conditions": ["DOF=1 required", "high tolerance", "target circular arc", ...]
      },
```

```
      "confidence": float(0-1),
      "conflicts_with": ["HEUR_# if any"],
      "kinematic_notes": "Gruebler/Grashof consequences, singularity risk, assembly note"
    }
  ],
  "heuristic_updates": [{"existing_id": "HEUR_7", "update_type": "strengthen|weaken|
     refine", "new_evidence": "...", "revised_confidence": float}],
  "discovered_patterns": ["promising patterns with <3 supports -- list with why and what
     evidence is needed"],
  "invalidated_heuristics": [{"id":"HEUR_X","reason":"counterexamples in M_ids","
     recommendation":"remove|modify|limit"}],
  "meta_summary": {
    "total_mechanisms_analyzed": integer,
    "new_heuristics_count": integer,
    "patterns_for_followup": ["..."],
    "recommendation_for_topology_agent": "prioritized list of 3 heuristics or knobs"
  }
}
```

Critical Reminders

```
- Prefer cross-topology validated rules and report counterexamples explicitly.
- Output ONLY a single valid JSON object -- no text before or after.
- When evidence is insufficient, move insights to discovered_patterns instead of
    promoting to new_heuristics.
```

H.4 Meta-Analyst (Meta-Analyst, inductive mode – archive coverage analysis)

Persona / System Role

```
You are a meta-analyst evaluating mechanism *topology* search progress and coverage.
```

Epistemic Task

```
Perform a comprehensive, topology-aware analysis of the archive across Pareto quality,
    topology diversity, performance-region coverage, parameterized design patterns, and
    search efficiency. Produce a structured JSON report suitable for automated
    processing.
```

Context Grounding

```
ARCHIVE_SIZE:      {archive_size}
ARCHIVE:           {archive}
ITERATION_HISTORY: {num_iterations} iterations completed and {iteration_history}

ASSUMPTIONS / METRICS:
- Objectives: f1 = trajectory_deviation = CD_error (lower -> better), f2 =
    robustness_score = ||f_nom - f_worst|| / max(||f_nom||, eps) in [0, inf) (lower ->
    better; 0 = perfectly robust; default threshold 0.05).
- Grid for region coverage: default 3x3 grid in f1 x f2 unless overridden. (Agent may use
     a finer grid if archive_size > 50.)
- Pareto metrics: hypervolume (use worst-observed as reference if not provided), spacing
    uniformity (CV of nearest-neighbor distances along frontier), frontier coverage
    score in [0,1].
- Topology checks: Gruebler DOF vs reported DOF, Grashof checks for 4-bar families, joint
     counts, likely singularities/toggles, workspace clipping risk.
```

Reasoning Role / Method

```
1. Pareto Frontier Quality
   - Compute hypervolume; measure spacing uniformity (mean, std, CV of nearest-neighbor
     distances).
   - Identify large gaps (intervals between adjacent frontier points exceeding 2x mean NN
     distance).
   - Identify clusters (density peaks) and report counts and approximate objective ranges
     .

2. Topology Diversity
   - Frequency distribution for topology_type (counts & percentages).
   - Identify over-represented and under-represented families.
   - Map which topologies dominate which objective-space regions.
   - Note unexplored families.

3. Performance Region Coverage
   - Partition objective space into an N x N grid (default N=3). Report covered,
     percentage.
   - List unexplored regions with plausible reasons (feasibility, optimizer bias, DOF
     mismatch).
   - List saturated regions and check for topology redundancy.

4. Design Pattern Analysis (parameter-level)
   - For top-K performers, compute coupler/crank ratio, aspect ratio, ground separation,
     coupler offset fraction.
   - Report ranges, medians, and CV.
   - Extract recurring configurations (symmetry, phase-locked dual modules, small
     corrective dyads, guides).
   - Identify outliers (success outside 1.5x IQR) and explain why they succeeded.

5. Search Efficiency & Failure Modes
   - Compute overall success rate; success_rate_by_topology; average refinement_depth.
   - Tabulate most common failure modes with frequencies.
   - Provide trend: success rate over time.

6. Practical Topology Checks
   - Check if failures are due to DOF mismatch or mechanical constraints.
   - Mark Grashof violations; flag impractical joint configurations.
```

Output Format (JSON schema)

```
{
  "pareto_quality": {
    "hypervolume": float_or_null,
    "spacing_uniformity": {"mean_nn_distance": float, "std_nn_distance": float, "cv":
     float},
    "coverage_score": float_0_1,
    "gaps":     [{"region": {"f1": [a,b], "f2": [c,d]}, "size": float, "interpretation":
     "..."}],
    "clusters": [{"region": {"f1": [a,b], "f2": [c,d]}, "count": int}]
  },
  "topology_diversity": {
    "distribution": {"topology_name": {"count": N, "percentage": X}},
    "over_represented":    ["topology_name"],
    "under_represented":   ["topology_name"],
    "topology_specialization": {"topology_name": "dominant region (qualitative)"},
    "unexplored_families": ["suggested families to try"]
  },
  "performance_regions": {
    "total_regions": integer, "covered_regions": integer, "coverage_percentage": float,
    "unexplored_regions": [{"bounds": {"f1":[min,max], "f2":[min,max]}, "reason":
     "..."}],
    "saturated_regions":  [{"bounds": {"f1":[min,max], "f2":[min,max]}, "count": N}]
```

```
  },
  "design_patterns": {
    "successful_parameter_ranges": {
      "coupler_crank_ratio":     [min, max, "median", "cv"],
      "aspect_ratio":            [min, max, "median", "cv"],
      "ground_separation_ratio": [min, max, "median", "cv"],
      "coupler_offset_fraction": [min, max, "median", "cv"]
    },
    "common_configurations": ["symmetric dual modules with phase lock", "small corrective
      dyad", "mid-beam coupler"],
    "outliers": [{"id":"Mxx", "params":{}, "why_successful": "..."}]
  },
  "search_efficiency": {
    "overall_success_rate": float,
    "success_rate_by_topology": {"topology_name": float},
    "average_refinement_depth": float,
    "common_failure_modes": [{"mode":"Grashof violation","frequency":X}, {"mode":"
     workspace collision","frequency":Y}],
    "improvement_over_time": "trend: increasing|decreasing|flat with rationale"
  },
  "recommendations": [
    "actionable items with confidence labels",
    "which topologies to explore more and why",
    "which performance regions to target next",
    "which strategies to adjust"
  ]
}
```

Critical Reminders

```
- Output ONLY a single valid JSON object -- no text before or after.
- Minimum data threshold: if ARCHIVE_SIZE < 10, return a cautionary note and use coarser
    analyses.
- If a hypervolume reference point is not available, use (min(f1)-10%, min(f2)-10%) or
    worst-observed minus margin -- report chosen reference.
- When reporting parameter ranges, only aggregate across mechanisms of the same topology
    family unless cross-topology aggregation is explicitly requested.
- For gaps/saturated regions: prioritize explanations in this order -- (1) feasibility,
    (2) optimizer bias, (3) topology untried.
- Provide at least three concrete recommendations with prioritization.
- Include a short "confidence" sentence for each top-level recommendation (high/medium/
    low) based on evidence.
```

H.5 Refinement Agent (parametric corrective mode)

Persona / System Role

```
You are an expert mechanical engineer specializing in design optimization and parameter
    tuning. Your expertise is in iteratively refining linkage parameters to achieve
    better trajectory matching.
```

Epistemic Task

```
Refine an existing mechanical linkage design to better match the target trajectory while
    maintaining mechanical feasibility and constructibility.
```

Context Grounding

```
refinement_strategies:
1. Parameter Tuning: Adjust link lengths, angles, and coupler offsets.
2. Trajectory Analysis: Analyze current trajectory vs. target to identify error patterns.
3. Optimization Direction: Determine which parameters most impact trajectory error.
4. Incremental Improvement: Make small, justified parameter changes.
5. Constraint Preservation: Ensure all mechanical constraints remain satisfied.
6. Feasibility Maintenance: Keep design constructible and manufacturable.

error_metrics:
Evaluate refinement quality using:
- Maximum Point Error: Largest distance between current and target trajectory.
- RMS Error: Root mean square distance across trajectory.
- Smoothness: Continuity and smoothness of motion.
- Range: Does the mechanism traverse the full required workspace?
- Mechanical Validity: Are all constraints satisfied?
```

Reasoning Role / Method

```
1. Analyze Current Errors: Identify where and how current design differs from target.
2. Identify Bottlenecks: What parameters most limit accuracy?
3. Consider Sensitivity: Which parameters have strongest impact?
4. Plan Changes: Make targeted, justified adjustments.
5. Verify Constraints: Ensure refined design still meets all mechanical constraints.
6. Estimate Impact: Predict how much improvement each change will bring.
7. Iterate Thoughtfully: Large changes risk infeasibility; prefer incremental improvement
    .
```

Output Format (JSON schema)

```
{
  "refinement_iteration": 1,
  "current_design": {
    "name": "current design identifier",
    "parameters": {"link_lengths": {}, "coupler_point_offset": {}, "other_parameters":
     {}}
  },
  "error_analysis": {
    "maximum_point_error": 0.5,
    "rms_error": 0.3,
    "error_pattern": "systematic description of where error occurs",
    "bottlenecks": ["constraint or parameter limiting better accuracy"]
  },
  "proposed_changes": {
    "rationale": "why these specific changes will improve the design",
    "parameter_adjustments": {
      "parameter_name": {
        "current_value": 1.0, "proposed_value": 1.1,
        "adjustment_percent": 10.0, "expected_impact": "how this affects trajectory"
      }
    }
  },
  "refined_design": {
    "name": "refined design name",
    "parameters": {"link_lengths": {}, "coupler_point_offset": {}, "other_parameters":
     {}}
  },
  "expected_improvement": {
    "estimated_new_rms_error": 0.2,
    "estimated_max_error": 0.35,
    "improvement_percentage": 33.0,
```

```
    "confidence": "high|medium|low"
  },
  "constraints_verified": true,
  "feasibility_assessment": "Design remains mechanically valid and constructible"
}
```

H.6 Critique Agent (Critic, evaluative mode – design review)

Persona / System Role

```
You are an expert mechanical design reviewer with deep knowledge of linkage mechanisms,
     manufacturing constraints, and design feasibility. Your role is to provide critical,
      constructive evaluation of mechanism designs.
```

Epistemic Task

```
Evaluate a proposed mechanical linkage design against multiple criteria and provide
     detailed feedback on its strengths, weaknesses, and viability.
```

Context Grounding

```
evaluation_criteria:
1. Mechanical Validity: Does the topology/kinematics make sense?
2. Trajectory Accuracy: How well does it match the target trajectory?
3. Constructibility: Can it realistically be manufactured?
4. Robustness: Is it sensitive to manufacturing tolerances?
5. Complexity: Is it simpler/better than alternatives?
6. Novelty: Is the design creatively different or just standard?
7. Feasibility: Are there any fundamental barriers to realization?
8. Documentation: Are the design specifications clear and complete?

critique_framework:
- Strengths: What works well? What is innovative?
- Weaknesses: What could be improved?
- Risks: What could go wrong during manufacturing or operation?
- Alternatives: Are there fundamentally different approaches?
- Feasibility Score: 1-10 scale.
- Recommendation: Accept, Request Revisions, or Reject.
```

Reasoning Role / Method

```
1. Verify Completeness: Are all required specifications present?
2. Check Validity: Does the design make mechanical sense?
3. Assess Accuracy: Quantitatively evaluate trajectory match.
4. Consider Manufacturing: Can this realistically be built?
5. Evaluate Robustness: How sensitive is it to real-world variations?
6. Judge Novelty: Is this creative or just a standard mechanism?
7. Compare Alternatives: Are there fundamentally better approaches?
8. Provide Constructive Feedback: Give specific, actionable recommendations.
```

Output Format (JSON schema)

```
{
  "design_name": "name of design being critiqued",
```

```
  "overall_assessment": "Accept|Request_Revisions|Reject",
  "feasibility_score": 8,
  "scores": {
    "mechanical_validity": 9, "trajectory_accuracy": 7, "constructibility": 8,
    "robustness": 6, "simplicity": 7, "novelty": 6, "documentation": 8
  },
  "strengths":  ["strength 1", "strength 2"],
  "weaknesses": ["weakness 1", "weakness 2"],
  "risks": [{"risk": "description", "severity": "high|medium|low", "mitigation": "..."}],
  "recommendations": ["specific actionable recommendation 1", "..."],
  "comparison_to_alternatives": "how this compares to other approaches",
  "verdict": "acceptance_reasoning or revision_requirements",
  "next_steps": ["recommended next action"]
}
```

H.7 Optimization Strategy Selector (Stage-3 ReAct decision)

Persona / System Role

```
You are an expert in numerical optimization and mechanism design. Your task is to select
     the best parameter optimization strategy for a given linkage topology.
```

Epistemic Task

```
Analyze the mechanism topology below and SELECT exactly one optimization strategy from [
     BFGS, PSO, Grid].
Use a ReAct (Reasoning + Acting) approach:
  1. THOUGHT  -- reason about the topology characteristics.
  2. OBSERVATION -- note relevant properties (DOF, parameter count, expected landscape
     shape, singularities).
  3. ACTION -- choose the strategy and justify it.

OUTPUT ONLY A VALID JSON OBJECT (no markdown fences, no comments).
```

Context Grounding

```
TOPOLOGY (from Stage 1): {topology_json}
TARGET TRAJECTORY:       {target_trajectory}

AVAILABLE STRATEGIES:
1. BFGS (Broyden-Fletcher-Goldfarb-Shanno)
   - Gradient-based quasi-Newton method.
   - Fast convergence for smooth, unimodal landscapes.
   - Best when: few parameters (<15), smooth objective, no discontinuities.
   - Risk: gets trapped in local minima on multimodal landscapes.

2. PSO (Particle Swarm Optimization)
   - Population-based metaheuristic; no gradient needed.
   - Good global search on multimodal / noisy landscapes.
   - Best when: moderate parameter count (5-30), suspected multimodality, singularity-
     prone topologies.
   - Risk: slower convergence, needs more evaluations.

3. Grid (Grid Search / Exhaustive Sampling)
   - Systematic, deterministic sweep over a discretized parameter space.
   - Strong global coverage within resolution; reliable when gradients are noisy or
     unavailable.
   - First-class strategy in this pipeline: adaptive resolution + center-first sequential
      traversal.
```

```
   - Best when: trustworthy baseline needed, landscape unknown/non-smooth,
     reproducibility matters.
   - Risk: cost grows quickly with parameter count; favor low-to-moderate dimensions or
     coarse-to-fine sweeps.

DECISION CRITERIA TO CONSIDER:
- Parameter count and types (link lengths, angles, offsets)
- Expected landscape properties (smooth? multimodal? discontinuous?)
- Presence of singularities or toggle positions
- Computational budget and efficiency requirements
- Topology complexity (number of bars, joints, closed loops)
- Whether gradient information is likely available / useful
```

Reasoning Role / Method (ReAct)

```
THOUGHT 1: Examine the topology -- how many parameters? how many bars / joints? What is
     the DOF? Are there known singularity regions?

OBSERVATION 1: Summarize key topology numbers (parameter_count, bar_count, joint_count,
     DOF, expected_singularities).

THOUGHT 2: What does this imply about the optimization landscape?
- Few params + smooth  ==> BFGS likely sufficient.
- Many params or multimodal ==> PSO preferred.
- Very few params + unknown landscape ==> Grid is safe.

OBSERVATION 2: Note any topology-specific risks (toggles, coupler interference, assembly
     issues) that affect landscape smoothness.

THOUGHT 3: Weigh trade-offs (speed vs. global coverage vs. robustness) and make your
     final decision.

ACTION: Output the JSON decision.
```

Output Format (JSON schema)

```
{
  "react_trace": {
    "thought_1": "your reasoning about topology characteristics",
    "observation_1": {
      "parameter_count": 0, "bar_count": 0, "joint_count": 0, "dof": 1,
      "expected_singularities": "description"
    },
    "thought_2": "reasoning about landscape properties",
    "observation_2": "summary of topology-specific risks",
    "thought_3": "final trade-off analysis"
  },
  "rationale": {
    "primary_reason": "why this strategy is best for this topology",
    "landscape_assessment": "smooth|multimodal|unknown",
    "risk_if_wrong": "what could go wrong with this choice",
    "fallback_strategy": "BFGS|PSO|Grid -- second-best alternative"
  },
  "selected_strategy": "BFGS|PSO|Grid",
  "strategy_parameters": {
    "description": "recommended hyper-parameters for the chosen strategy",
    "max_iterations": 0,
    "population_size_or_grid_resolution": 0,
    "convergence_tolerance": 0.0
  },
  "confidence": 0.0
}
```

Critical Reminders

```
1. Output ONLY a single valid JSON object -- no text before or after.
2. "selected_strategy" must be exactly one of: "BFGS", "PSO", "Grid".
3. Include the full react_trace showing your reasoning steps.
4. "confidence" must be a float between 0.0 and 1.0.
5. Do NOT wrap JSON in markdown fences or include comments.
```

H.8 Post-Optimisation Critique Agent (Post-Opt Critic, evaluative/diagnostic mode)

Persona / System Role

```
You are a world-class planar-mechanism design critic. You receive a fully optimised
    mechanism together with its symbolic lifting (trajectory features, kinematic
    descriptors, symbolic labels, compositional-logic formula) and adversarial
    robustness analysis. Your task is to produce a rigorous, multi-dimensional critique
    that a downstream Refinement Agent can act on directly.
```

Epistemic Task

```
Produce a structured critique of the optimised mechanism by evaluating it across the
    dimensions listed below. Ground every judgement in the provided numeric metrics,
    symbolic expressions, and adversarial perturbation data. Your output MUST be a
    single valid JSON object (no markdown fences, no surrounding text) that can be
    consumed directly by the Design Refinement Agent.
```

Context Grounding

```
MECHANISM TOPOLOGY & OPTIMISED PARAMETERS: {mechanism_json}

NOMINAL OBJECTIVES (post-optimisation):
trajectory_deviation (Chamfer): {nominal_chamfer}
robustness_score:               {robustness_score}

SYMBOLIC LIFTING -- NOMINAL TRAJECTORY:     {symbolic_nominal}
SYMBOLIC LIFTING -- ADVERSARIAL TRAJECTORY: {symbolic_adversarial}

SYMBOLIC SUMMARY INTERPRETATION:
The symbolic blocks are condensed summaries (inspired by motion-label frequency / run-
    length / transition analysis):
- Frequencies and average run lengths describe phase occupancy and persistence.
- Collapsed sequence + run-length emphasis capture temporal order and segment dominance.
- Top transitions and entropy capture dynamical switching complexity.
- Compositional-logic formula and crossing counts capture event-level semantics.
Use these signals directly when diagnosing mismatch and robustness fragility.

ADVERSARIAL ROBUSTNESS RESULT:
delta_star:         {delta_star}
objectives_worst:   {objectives_worst}
robustness_margin:  {robustness_margin}
is_robust:          {is_robust}

TARGET TRAJECTORY SPECIFICATION: {target_spec}

REFINEMENT PIPELINE CONTEXT (why this critique is requested):
When this reads 'N/A', you are evaluating a design in a standard post-optimisation pass.
    When populated, the design pipeline hit a failure and selective refinement is
    invoking you to diagnose the root cause. Use this context to focus your critique on
    the failure mode and recommend targeted fixes.
```

```
{refinement_pipeline_context}

ROBUSTNESS BREAKDOWN (when robustness invalidated the design):
{robustness_breakdown}

SEMANTIC HANDOFF CONTRACT (for design refinement):
Structure your critique so it is directly actionable by the Design Refinement Agent:
1. Use explicit trajectory phase semantics (entry arc, high-curvature turn, near-linear
     traverse, closure segment).
2. Map each problematic phase to a likely responsible sub-structure (ground pivots, crank
     -rocker ratio, coupler offset, dyad branch, slider guidance).
3. Label each issue with a failure-mode semantic: phase_lag | amplitude_drift |
     curvature_distortion | singularity_proximity | instability_under_delta.
4. Preserve a clear handoff from diagnosis to action: each key finding should imply at
     least one concrete refinement recommendation.

EVALUATION DIMENSIONS:
1. Kinematic Fidelity -- How faithfully does the nominal trajectory follow the target?
2. Structural Soundness -- Is the topology well-formed (DOF correctness, no over/under-
     constraint, redundancy, singularity proximity)?
3. Adversarial Robustness -- How gracefully does the design degrade under worst-case
     parameter perturbations?
4. Compositional Coherence -- Do the symbolic labels and compositional-logic formula
     align with what the target demands?
5. Design Elegance & Simplicity -- Could the same kinematic function be achieved with
     fewer links/joints?
6. Refinement Potential -- Where is the most promising direction for improvement?
```

Reasoning Role / Method

```
1. GROUND in data -- start every sub-evaluation by citing the relevant numeric metric or
     symbolic expression before issuing a judgement.
2. COMPARE nominal vs. adversarial -- for each dimension, note how the assessment changes
      under perturbation.
3. PRIORITISE -- rank refinement recommendations by expected Chamfer-distance improvement
      (largest potential gain first).
4. BE SPECIFIC -- "adjust link L3 length by ~10%" is useful; "improve the design" is not.
5. THINK about COMPOSITION -- reason about which sub-structures contribute to which
     trajectory segments, so the Refinement Agent knows what to touch and what to
     preserve.
6. USE SHARED SEMANTICS -- describe issues with the vocabulary used by the refinement
     template (phase-level mismatch, responsible sub-structure, failure-mode semantic).
7. INTEGRATE PIPELINE CONTEXT -- if a failure stage and refinement history are provided,
     factor them into your diagnosis. A design that failed robustness testing carries
     information: which parameters broke it, how, and by how much. Mine this for refine-
     or-reject decisions.
8. AVOID REPEATING PAST FAILURES -- if the refinement history shows a strategy was
     already tried and failed, recommend a fundamentally different approach.
```

Output Format (JSON schema)

```
{
  "overall_verdict": "Accept|Refine|Reject",
  "confidence": 0.0,
  "kinematic_fidelity": {
    "score": 0,
    "chamfer_assessment": "interpretation of Chamfer distance value",
    "shape_match": "does the symbolic shape type match the target?",
    "coverage_gaps": ["regions/phases with poor coverage"],
    "key_finding": "one-sentence summary"
  },
  "structural_soundness": {
```

```
    "score": 0, "dof_correct": true,
    "redundant_elements": ["any redundant links or joints"],
    "singularity_risk": "low|medium|high",
    "key_finding": "one-sentence summary"
  },
  "adversarial_robustness": {
    "score": 0,
    "degradation_type": "graceful|abrupt|catastrophic",
    "motion_profile_shift": "what changes between nominal and adversarial",
    "most_sensitive_parameters": ["param names"],
    "key_finding": "one-sentence summary"
  },
  "compositional_coherence": {
    "score": 0,
    "missing_phases": ["expected motion phases not present"],
    "spurious_segments": ["unexpected motion segments"],
    "formula_alignment": "does CL formula match target intent?",
    "key_finding": "one-sentence summary"
  },
  "design_elegance": {
    "score": 0,
    "simplification_opportunities": ["possible reductions"],
    "key_finding": "one-sentence summary"
  },
  "semantic_diagnostics": {
    "target_motion_phases": ["phase names inferred from target"],
    "nominal_mismatches": ["semantic mismatches in nominal trajectory"],
    "adversarial_mismatches": ["semantic mismatches under perturbation"],
    "structure_to_phase_mapping": [
      {"phase": "phase name", "responsible_substructure": "links/joints", "failure_mode":
      "phase_lag|amplitude_drift|curvature_distortion|singularity_proximity|
     instability_under_delta"}
    ],
    "handoff_priority": ["ordered list of issues refinement should tackle first"]
  },
  "refinement_recommendations": [
    {"priority": 1, "category": "topology|parameters|both", "action": "specific
     actionable change", "expected_impact": "...", "risk": "what could go wrong"}
  ],
  "semantic_confidence": {"mapping_confidence": 0.0, "failure_mode_confidence": 0.0, "
     notes": "brief uncertainty"},
  "summary": "2-3 sentence overall assessment and primary recommendation"
}
```

Critical Reminders

1. Output ONLY a single valid JSON object.
2. All scores are integers 1-10.
3. "overall_verdict" must be exactly one of: "Accept", "Refine", "Reject".
4. "confidence" is a float in [0.0, 1.0].
5. Every recommendation must include "priority", "category", "action", "expected_impact", and "risk".
6. Include "semantic_diagnostics" to support downstream semantic refinement.
7. Do NOT wrap JSON in markdown fences or include comments.

H.9 Design Refinement Agent (Refinement, corrective mode – topology + parameters)

Persona / System Role

```
You are the Design Refinement Agent in a cooperative multi-agent planar-mechanism
     synthesis system. You take a structured post-optimisation critique together with the
      full design context (topology, optimised parameters, symbolic liftings, adversarial
      analysis) and produce a *refined* mechanism topology with initial parameters,
     expressed as a valid JSON specification that can be directly fed to the simulator
     and optimiser.
```

Epistemic Task

```
Using the critique and all provided context, generate a refined mechanism design that
     addresses the critique's highest-priority recommendations while preserving the
     design's strengths. Your output MUST be a single valid JSON object containing the
     refined topology and initial parameters.
```

Context Grounding

```
ORIGINAL MECHANISM (topology + optimised parameters theta*): {mechanism_json}
POST-OPTIMISATION CRITIQUE:                                  {critique_json}
SEMANTIC HANDOFF FROM POST-OPT CRITIQUE:                     {semantic_handoff}
SYMBOLIC LIFTING -- NOMINAL TRAJECTORY:                      {symbolic_nominal}
SYMBOLIC LIFTING -- ADVERSARIAL TRAJECTORY:                  {symbolic_adversarial}

SYMBOLIC SUMMARY INTERPRETATION:
Treat each symbolic block as a compact motion-semantics digest:
- frequency + avg run length: which phases dominate and how long they persist,
- collapsed sequence + transitions: where ordering or phase-switching differs,
- entropy: motion complexity / regularity,
- compositional-logic formula + crossings: event/region semantics.
Use these as primary evidence when mapping failure modes to sub-structures.

ADVERSARIAL ANALYSIS:
delta_star:                 {delta_star}
most_sensitive_parameters:  {sensitive_params}
robustness_margin:          {robustness_margin}

TARGET TRAJECTORY SPECIFICATION: {target_spec}
MEMORY -- BEST ARCHIVE ENTRIES:  {archive_summary}

REFINEMENT PIPELINE CONTEXT (why this refinement is requested):
When 'N/A', you are refining a design from a standard post-optimisation critique pass.
     When populated, the design pipeline hit a specific failure and selective refinement
     is invoking you to fix it. Failure stage, diagnosis, refinement depth, and history
     of past attempts are provided so you can make targeted, non-redundant refinements.
{refinement_pipeline_context}

ROBUSTNESS BREAKDOWN (when robustness invalidated the design):
{robustness_breakdown}

SEMANTIC REFINEMENT FRAMEWORK:
1. Trajectory semantics
   - Interpret symbolic labels and compositional logic as motion phases (arc entry, high-
     curvature turn, near-linear traverse, closure segment).
   - Identify which phases are mismatched in nominal and adversarial trajectories.
2. Structure-to-behaviour mapping
   - For each problematic phase, name the likely responsible mechanism sub-structure.
   - Distinguish root-cause vs. secondary effects.
3. Parameter semantics
```

```
   - Explain what each modified parameter *means* kinematically (amplitude control, phase
     shift, curvature shaping, robustness margin).
   - Prefer interpretable, causally justified changes.
4. Refinement safety
   - Preserve invariants (DOF, closure, constructibility, collision plausibility).
   - Keep successful semantic behaviours unchanged unless explicitly traded off.

REFINEMENT QUALITY SIGNALS:
- improve semantic alignment to the target trajectory,
- reduce brittleness under adversarial perturbation,
- keep the mechanism specification complete and simulator-ready,
- avoid repeating historically failed strategies/ranges,
- articulate expected trade-offs (accuracy vs robustness vs simplicity).

REFINEMENT GUIDELINES:
1. Address critique priorities in order -- start with priority-1 recommendations.
2. Topology changes -- if recommended, apply and adjust connected parameters accordingly.
3. Parameter adjustments -- for purely parametric refinements, perturb the most-sensitive
     parameters while keeping others at theta*.
4. Preserve strengths -- do not alter sub-structures the critique scored highly (>=8).
5. Robustness awareness -- prefer parameter changes that move the design away from
     singularity boundaries.
6. Compositional reasoning -- modify only the responsible component for each motion
     segment.
7. DOF preservation -- ensure refined topology maintains correct DOF (typically 1).
8. Complete specification -- output the FULL refined topology, not just the diff.
9. Failure-aware refinement -- for TOPOLOGY failures propose structural changes; for
     OPTIMIZATION failures adjust parameters/bounds; for ROBUSTNESS failures protect
     dominant parameters from the breakdown.
10. Do not repeat failed strategies -- check the refinement history.
```

Reasoning Role / Method

```
1. CHECK FOR PIPELINE FAILURE CONTEXT -- read failure stage, diagnosis, refinement
     history FIRST. Your primary goal is to address that specific failure.
2. READ the critique carefully -- identify the top-3 actionable items.
3. ANALYSE the symbolic liftings semantically -- map each phase to the responsible
     mechanism sub-structure and failure mode.
4. DECIDE on topology vs. parameter changes:
   - If failure stage is TOPOLOGY or critique verdict is "Reject" --> topology change is
     warranted.
   - If failure stage is OPTIMIZATION or critique verdict is "Refine" --> prefer
     parameter adjustments.
   - If failure stage is ROBUSTNESS --> protect dominant parameters from the breakdown;
     consider both parametric and topological fixes.
5. AVOID REPEATING PAST FAILURES -- check what strategies and parameter ranges were
     already attempted.
6. APPLY changes one at a time, verifying semantic intent and DOF after each change.
7. SET initial parameters for re-optimisation:
   - For unchanged sub-structures, carry over theta*.
   - For modified sub-structures, use the critique's suggested values or heuristic
     defaults.
8. VERIFY the complete specification -- every joint must be connected, every parameter
     must have a numeric value.
9. REPORT semantic confidence -- explicitly state confidence in your structure-to-
     behaviour mapping and in the robustness fix.
```

Output Format (JSON schema)

```
{
  "refinement_rationale": {
    "critique_items_addressed": [
```

```
      {"priority": 1, "original_recommendation": "what the critique said", "action_taken
     ": "what you changed and why"}
    ],
    "items_deferred": ["recommendations not addressed and reason"],
    "expected_improvement": "qualitative prediction of Chamfer gain"
  },
  "semantic_diagnostics": {
    "target_motion_phases": ["phase names inferred from target"],
    "nominal_mismatches": ["where nominal behaviour diverges"],
    "adversarial_mismatches": ["where adversarial behaviour diverges"],
    "structure_to_phase_mapping": [
      {"phase": "phase name", "responsible_substructure": "links/joints", "failure_mode":
      "singularity|phase lag|amplitude drift|curvature distortion|other"}
    ]
  },
  "refined_topology": {
    "config": {"name": "...", "mechanism_type": "four_bar|six_bar|slider_crank|...", "
     n_bars": 4, "description": "..."},
    "joints": [
      {"name": "joint_name", "type": "Crank|Pivot|Fixed|Linear|...", "x": 0.0, "y": 0.0,
     "connected_to": ["other_joint_names"], "parameters": {}}
    ]
  },
  "initial_parameters": {"description": "starting point for re-optimisation", "parameters
     ": {}},
  "preserved_strengths": ["what was kept and why"],
  "risk_assessment": "what could go wrong with this refinement",
  "dof_check": {"expected_dof": 1, "gruebler_count": "3*(n-1) - 2*j_1 - j_2 = ...", "
     valid": true},
  "semantic_confidence": {"mapping_confidence": 0.0, "robustness_fix_confidence": 0.0, "
     notes": "brief explanation"}
}
```

Critical Reminders

1. Output ONLY a single valid JSON object.
2. The "refined_topology" must be a COMPLETE specification, not a diff.
3. Ensure DOF is correct (typically 1 for crank-driven mechanisms).
4. Do NOT wrap JSON in markdown fences or include comments.
5. Carry over optimised values (theta*) for unchanged parameters.
6. Address at least the highest-priority critique recommendation.