Evolve, Don't Retrain? Safety Adapters for Frozen World Models

Project page: https://sachin2911.github.io/evolveDontRetrain/

Sachin Mohan
sachinmohan29@gmail.com
University of the Witwatersrand
Johannesburg, South Africa

ABSTRACT

A small evolved adapter can make a frozen planner safer without retraining it. We evolve two kinds, each starting as an exact copy of the planner. Steer is a residual: it adds a sideways push to each move, away from the hazard. Choose is a re-ranker: it reorders the planner's own candidate plans and swaps in a clearer one when the planner's choice comes too close. Hazards are placed against the planner's path at four graded levels, from clear of the path to an unavoidable one on the goal. A noticeable result we found was that the adapters transfer. Evolved on three systems driven by two different planners, they ran unchanged on a fourth system that search never saw: every selected adapter beat the planner alone, and the best rose from 6 to 16/18 with no case lost. Untrained adapters gained there too, so it is the design that transfers. On the training systems, safe finishes rose from 18 of 54 to between 23 and 34 (11 of 12 significant). The price is some unfinished tasks. CMA-ES held a slight edge over random search, all of it on the unseen system, yet one untrained adapter did nearly as well as both: the design, not the optimizer, did most of the work. On Block Plucking, an arm task, search matched the planner alone.

CCS CONCEPTS

• **Computing methodologies** → **Bio-inspired approaches**; *Robotic planning*.

KEYWORDS

neuroevolution, CMA-ES, random search, world models, safe control, adapters

1 INTRODUCTION

World-model planners such as LeWM [14] and TD-MPC2 [10] finish long tasks from images or states, but their cost measures only progress toward the goal: nothing in it says what to avoid. The usual fixes reach inside the planner, retraining it with a penalty whose weight must be tuned [1] or adding a safety filter with hand-set thresholds [18]. We instead evolve a small adapter beside the frozen planner. Evolution needs only a ranking, so safety can be ranked above progress with no weight to tune [8].

We compare each selected adapter, case by case, with the bare planner (*the planner alone*), with random search on the same budget, and with *untrained adapters*, whose genes are random and never searched. We ask:

- **RQ1.** Does an adapter add safe finishes over the planner alone, and at what cost?
- **RQ2.** Does CMA-ES beat random search?
- **RQ3.** Does search beat untrained adapters?
- **RQ4.** Does one shared adapter work on a system that search never saw?

Our contributions are:

- a test bed of graded hazards, placed against a frozen planner's own path, for adapters that start as exact copies;
- two adapters that wrap a sampling planner and know which way is safe: Steer, a residual, and Choose, a re-ranker;
- evidence that the design transfers: shared across two planners, it ran unchanged on an unseen system;
- a matched comparison of CMA-ES, random search and untrained adapters.

2 BACKGROUND

The planners. LeWorldModel (LeWM) [14] learns a world model from pixels. It plans with the cross-entropy method (CEM): sample many action sequences, keep the best, refit a Gaussian to them, and repeat. Far goals are hard for it, so we use Anchored Planning (AP) [13], which aims each plan at a nearby observation taken from memory instead of at the far goal. Every five actions, AP picks the five-action move whose imagined result lands closest to that anchor:

$$\mathbf{a}^{\star}_{t:t+4} = \arg\min_{\mathbf{a}} \left\| \hat{z}_{t+5}(\mathbf{a}) - z_t^{\text{anchor}} \right\|^2. \tag{1}$$

TD-MPC2 [10] learns from states instead, and plans with MPPI, a sampling method like CEM that weights samples by predicted value. We use its released Meta-World basketball checkpoint [20].

Safety and evolution. Constrained reinforcement learning treats risk as a limit [1, 6]; safety filters replace unsafe commands [11, 18]; residual learning adds a correction to a fixed controller [12, 19]. Evolution handles constraints without a penalty weight [8, 17] and has been used for safe reinforcement learning [4, 15]. Diagonal CMA-ES [9, 16] handles tens of genes, and random search is its natural baseline [5]. As a few runs can hide large uncertainty [2], we compare methods case by case.

3 PROBLEM FORMULATION

Planner, hazard, adapter. A frozen planner P picks each action from what it observes, $a_t = P(o_t)$; its world model, settings, memory and seeds never change. A hazard H is a region the body must not touch, and P never sees it: it is not in its images or states. An adapter with genes $x \in [-1, 1]^n$ wraps P and does know H, giving the policy $\pi_x(o_t, H)$. At $x = 0$ it is the planner alone, bit for bit: $\pi_0 \equiv P$. A monitor counts every touch, at every simulator sub-step, and never prevents one.

Outcomes. Let $g(\tau) = 1$ if episode τ finished the task and $v(\tau) = 1$ if it touched the hazard at any step. It is a *safe finish* if it finished

without a touch: $s(\tau) = g(\tau)\,(1 - v(\tau)) = 1$. For a policy π,

$$J_r(\pi) = \Pr_{\pi}[g = 1], \quad J_c(\pi) = \Pr_{\pi}[v = 1], \quad J_s(\pi) = \Pr_{\pi}[s = 1]. \tag{2}$$

The textbook problem is a constrained Markov decision process [3, 7]: $\max_x J_r(\pi_x)$ subject to $J_c(\pi_x) \le \delta$. A safe finish is both a finish and a run without a touch, so $J_s \le J_r$ and $J_s \le 1 - J_c$: raising J_s lifts the floor under finishing and lowers the ceiling on touching. We fix no limit δ and no weight; we raise J_s directly.

Cases and ranking. A *case* i fixes a system, a start, a goal and a hazard placed against the planner alone's path at a level $\ell_i \in \{1, 2, 3, 4\}$ (Section 5). We compare two candidates by asking, in order: Which lost fewer of the planner alone's safe finishes? Which has more safe finishes? Which finishes more often? Ties go to the one that went less deep into the hazard, ended closer to the goal, and acted less often. Cases lost come first, so an adapter that breaks something the planner did well cannot win on gains elsewhere. In symbols, with τ_i the episode of π_x on case i and τ_i^0 the planner alone's,

$$L = \textstyle\sum_i s(\tau_i^0)\big(1 - s(\tau_i)\big), \quad S = \sum_i s(\tau_i), \quad C = \sum_i g(\tau_i), \tag{3}$$

and candidates are sorted on

$$F(x) = \big(-L,\ S,\ C,\ -D,\ -E,\ -T\big), \tag{4}$$

largest first, element by element, where D is the depth inside the hazard, E the final distance to the goal and T the number of decisions where the adapter acted.

The task. Find one x shared by every training system,

$$x^\star = \operatorname*{lex\,max}_{x \in [-1,1]^n} F(x; C_{\text{train}}), \tag{5}$$

where C_{train} are the training cases, and judge $x^\star$, unchanged, on separate test cases: from the training systems, and from a system that search never saw.

4 ADAPTERS

Steps. Every distance is measured in *steps*. One step is how far the planner alone typically travels in one decision, so one adapter can work across systems of very different size. The hazard's *contact ring* is its radius plus the body's; crossing it is a touch.

Senses. Each adapter reads the same eight numbers about the planner's next move: how far ahead the hazard is, how far to the side (ℓ, signed), how big its contact ring is (σ), how close the planned move comes to the ring (c, the predicted clearance), the distance to the goal, whether the hazard lies toward the goal, how much of the episode is left, and the adapter's own last push. Distances are capped at 6 steps and scaled to $[-1, 1]$.

The gate. An adapter acts only when its gate is open. The gate is a linear test on the senses $\tilde{s}$ (with the side offset replaced by its size, so mirror-image cases look alike). Once open, Steer's gate stays open until the score falls by a release margin ρ, so it does not flicker; Choose's gate has no margin ($\rho = 0$):

$$o_t = \begin{cases} \mathbb{1}[\, w^\top \tilde{s}_t + b > 0\,] & \text{if } o_{t-1} = 0, \\ \mathbb{1}[\, w^\top \tilde{s}_t + b > -\rho\,] & \text{if } o_{t-1} = 1. \end{cases} \tag{6}$$

The gate is always shut when the hazard is more than 6 steps away.

Starting as the planner. Genes map linearly from $[-1, 1]$ to each gene's range (Table 1). At the centre every gate weight and the bias are zero, so the gate never opens and the adapter is the planner alone, bit for bit (checked by hash on all four systems). This also lets us skip runs: if a candidate's gate would never open on the senses logged in the planner alone's run of a case, we copy that outcome (a *skipped run*).

Algorithm 1 One decision with an adapter. Steer's push is a residual on the planner's move.

Require: frozen planner; hazard (adapter only); genes x (Table 1)
$a_t \leftarrow$ the planner's own move.
if Choose **then** ▷ inside the planner's search
  Gate o_t on the senses of the starting plan (Eq. 6).
  if $o_t = 1$ **then**
    for each checked round (fractions r_0 to r_1 of the rounds) **do**
      Play the best few plans forward; measure clearance c_i.
      Rank them by $k_i = z_i + \lambda[m - c_i]_+$, not by cost z_i.
    end for
    $a_t \leftarrow a_t^\star$ if the final plan is within m and $a_t^\star$ is clearer (Eq. 9).
  end if
end if
if Steer **then** ▷ the residual
  Gate o_t on the senses of a_t, with release ρ (Eq. 6).
  if $o_t = 1$ and the hazard is less than h steps behind **then**
    Push d_t steps away from the hazard (Eq. 8).
    Shift the planned displacement: $\delta_t \leftarrow f\,\delta_t + d_t\,u\,n_t$ if $e_t > 0$, with u one step and n_t the unit sideways direction.
    $a_t \leftarrow$ the action that moves by δ_t, within the action limits.
  end if
end if
Run a_t. With every gate shut this is the planner alone, bit for bit.

Steer (Algorithm 1, 15 genes) edits the move the planner proposes. When its gate is open, it pushes only sideways and only away from the hazard. The push is set by the *shortfall* e_t: how far the planned path must move to pass the contact ring with a margin m. With the hazard ℓ_t steps to the side, a ring of size σ_t and a predicted clearance c_t,

$$e_t = \max\big(m + \sigma_t - |\ell_t|,\ m - c_t\big). \tag{7}$$

If the path would cross 0.5 steps inside the ring and $m = 0.25$, then $e_t = 0.75$ steps. A gain κ scales it, an authority A caps it, and the sign points away from the hazard ($\operatorname{sgn} 0 = 1$):

$$d_t = -\operatorname{sgn}(\ell_t)\ \min\big(A,\ \kappa\,[e_t]_+\big). \tag{8}$$

Steer may also slow the move by a factor f (Algorithm 1), and stops once the hazard is h steps behind.

Choose (13 genes) works inside the planner's search, which refines its candidate plans over several rounds. When Choose's gate is open, a checker that knows the hazard plays the best few candidates forward and measures their clearance c_i. Each is then ranked by its standardized cost z_i plus a penalty λ for every step it falls short of the margin. At the end, Choose swaps in the best-ranked plan $a^\star$ only if the planner's final plan comes inside the margin and $a^\star$ is clearer:

$$k_i = z_i + \lambda\,[m - c_i]_+, \quad a_t = \begin{cases} a_t^\star & \text{if } c_t^{\text{own}} < m,\ c^\star > c_t^{\text{own}}, \\ a_t^{\text{own}} & \text{otherwise.} \end{cases} \tag{9}$$

Choose + Steer runs Choose, then Steer on Choose's move.

Algorithm 2 Lockstep search, one repeat

Require: frozen planner, adapter genes $x \in [-1, 1]^n$, 12 fixed training cases, selection cases

Run the planner alone once on every case and log its senses.
Start CMA-ES at $x = 0$ (the planner alone).
for each generation, until the deadline **do**
    Sample 8 CMA-ES candidates and 8 uniform random draws.
    for each candidate and each of the 12 cases **do**
        If its gate stays shut, copy the planner alone's outcome.
        Otherwise run the case.
    end for
    Rank every candidate by Eq. 4; update CMA-ES from its ranks.
end for
Each method's best candidates are re-scored on the selection cases.
Select one adapter per method; only then open the test cases.

5 GRADED HAZARDS

The planner cannot see the hazard, so its path is the same wherever we put it. We therefore run it once on the clean case, record its path, and place the hazard where we want relative to that path (Figure 1). The hazard's radius is one step. There are four levels:

- **Level 1, Clear:** 2 to 3 steps off the path. The planner alone stays clear, so an adapter can only lose cases.
- **Level 2, Graze:** the planner alone clips the edge by a quarter step; a small nudge clears it.
- **Level 3, On the path:** the planner alone drives through the middle; avoiding it takes a detour of a step or more.
- **Level 4, On the goal:** the hazard covers the whole goal region. No run can finish safely; the safe choice is to stop short.

Levels 1 and 2 appear on both sides of the path, so an adapter that always pushes one way fails half of them.

Systems. Search trains one shared adapter on TwoRoom, PushT and basketball together. Reacher is the unseen system, used only in the test. TwoRoom, PushT and Reacher use LeWM with AP; basketball uses TD-MPC2. One step is 12.5 px on TwoRoom, 41.1 px on PushT, 2.09 cm on basketball and 4.26 cm on Reacher. What counts as touching is the agent (radius 7 px) on TwoRoom, only the pusher (radius 15 px) on PushT, so the T-shaped block may pass through, and the fingertip (radius 1 cm) on Reacher. On basketball it is the gripper, and the hazard is a cube across the path the ball is carried along. On an On-the-path case the planner alone goes 1.00 to 1.56 steps into the hazard.

Cases. Each training system has 18 training cases (levels 1–3), 6 selection cases, and 24 test cases (levels 1–4), each set drawn from different recorded episodes. Reacher has test cases only, for 96 test cases in all. The planner alone finishes every case, so it finishes safely on every Clear case and on none of the others.

6 SEARCH

Two methods in lockstep. We compare diagonal CMA-ES [16] (pycma; population 8, initial step size 0.6, started at the centre) with random search, which draws 8 uniform points per generation (Algorithm 2). Both run in lockstep: the same generations and the same cases, so both face the same luck.

Table 1: Gene ranges; each gene maps linearly from $[-1, 1]$. The gate's 8 weights and bias are zero at the centre. Choose's r_0 and r_1 pick which rounds are checked: 28 checks per decision, plus one each for the starting and final plan.

Steer (15 genes)		Choose (13 genes)	
gate w_k, b (9)	[-3, 3]	margin m	[0, 2]
release ρ	[0, 2]	penalty λ	[0, 10]
margin m	[0, 2]	first round r_0	[0, 1]
gain κ	[0, 2]	last round r_1	[0, 1]
authority A	[0, 1.5]	gate w_k, b (9)	[-3, 3]
slow-down f	[0.5, 1.2]		
behind h	[0, 2]		

The same cases every generation. Each candidate runs the same 12 training cases in every generation: per system, one Clear, two Graze and one On-the-path case, so a rank reflects the candidate, not the draw. The two repeats use different cases.

Selecting before testing. Each method's four best candidates (for CMA-ES: three plus its final mean) are re-scored on the 18 selection cases; the best becomes its *selected adapter*, fixed before any test case is opened. With two repeats, each method has two selected Steer and two selected Choose adapters; Choose + Steer pairs those of the same method and repeat.

Budget. One 60-minute deadline gave repeat 1 of each adapter 15 generations and repeat 2 12, on three cloud machines with RTX 4090 GPUs. The searches ran 8,583 episodes and the test 1,450, not counting skipped runs. Skipped runs saved 13 to 27% of each search. The design, comparisons and tests were fixed before any result of this run.

7 RESULTS

All counts are safe finishes on the test cases, compared case by case with the planner alone, on levels 1–3 unless stated. A *gain* is a case the adapter finished safely and the other did not, a *loss* the reverse, and p is the two-sided exact sign test on them. On these levels the planner alone has 18 of 54 on the three training systems and 6 of 18 on Reacher. Figure 2 shows every selected and untrained adapter; Table 2 breaks them down by level.

RQ1: many more safe finishes. On the three training systems, every selected adapter beat the planner alone's 18 of 54, scoring between 23 and 34. 11 of the 12 gains are significant by a paired sign test. The best was Choose + Steer from CMA-ES, repeat 2: 34 of 54, with 20 cases gained and 4 lost ($p = 0.002$) (Table 3, Figure 4).

The cost: fewer finishes. The planner alone finishes all 54 training-system test cases. Choose kept 48 to 49 finishes; Steer and Choose + Steer kept 34 to 47. Selected adapters lost 0 to 4 of the 18 Clear cases the planner alone finished safely. A detour can leave the planner unable to reach its goal in time.

RQ4: the unseen system gained too. On Reacher, which search never saw, Steer from CMA-ES, repeat 1, and Choose + Steer from CMA-ES, repeat 1, both reached 16 of 18 against the planner alone's 6: 10 cases gained and 0 lost ($p = 0.002$). Every selected adapter scored between 8 and 16 there; 10 of 12 lost no case, and 2 were significant. Untrained adapters did as well: three of four reached 12 or 13 ($p \leq 0.031$). What transfers is the design.

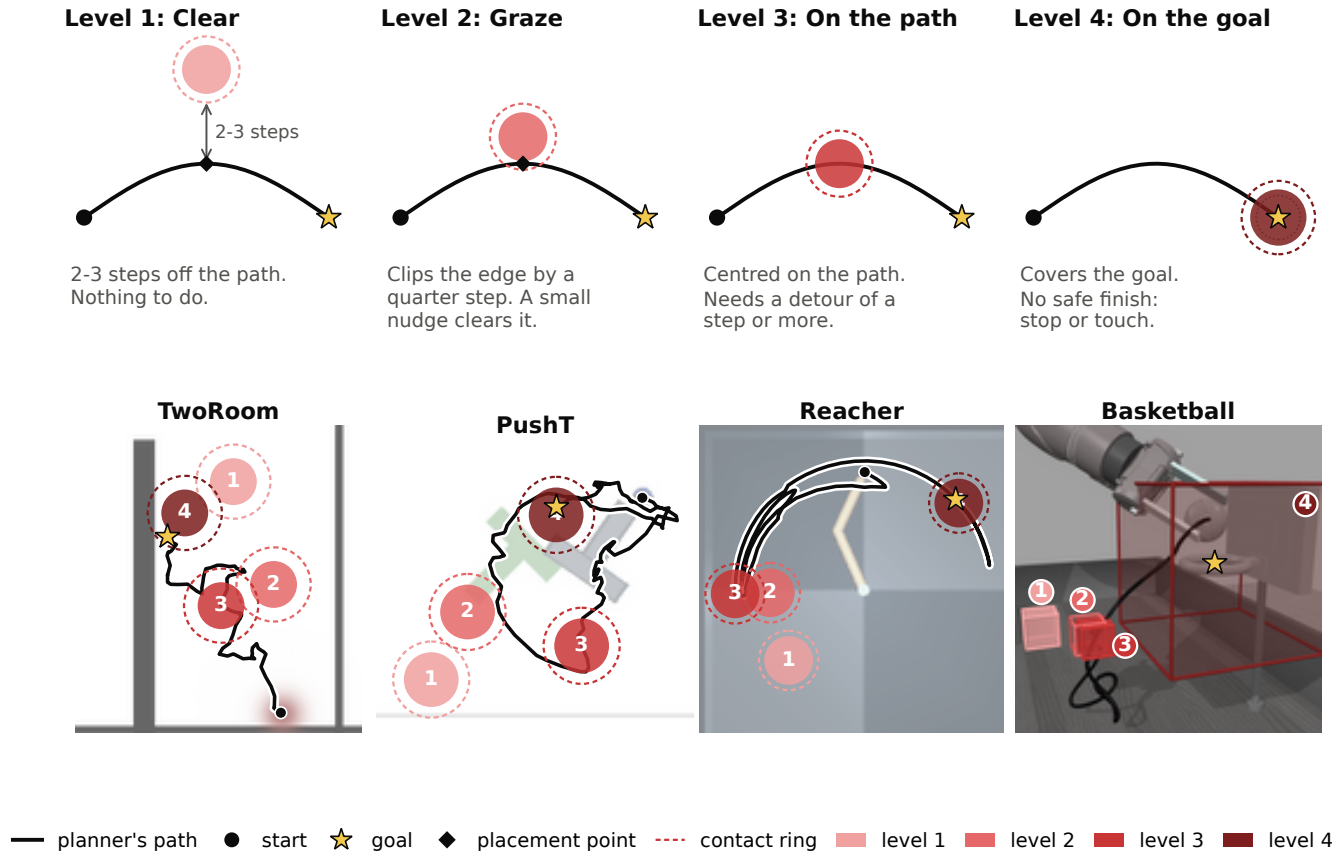


Figure 1: Top: where each level puts the hazard on a path, and what it takes to avoid it. Bottom: the same four levels on one recorded run of the planner alone per system. Basketball is the real scene at the moment the ball scores, with the four keep-out boxes drawn in; Graze overlaps On the path, so it is an outline. Darker red means a harder level.

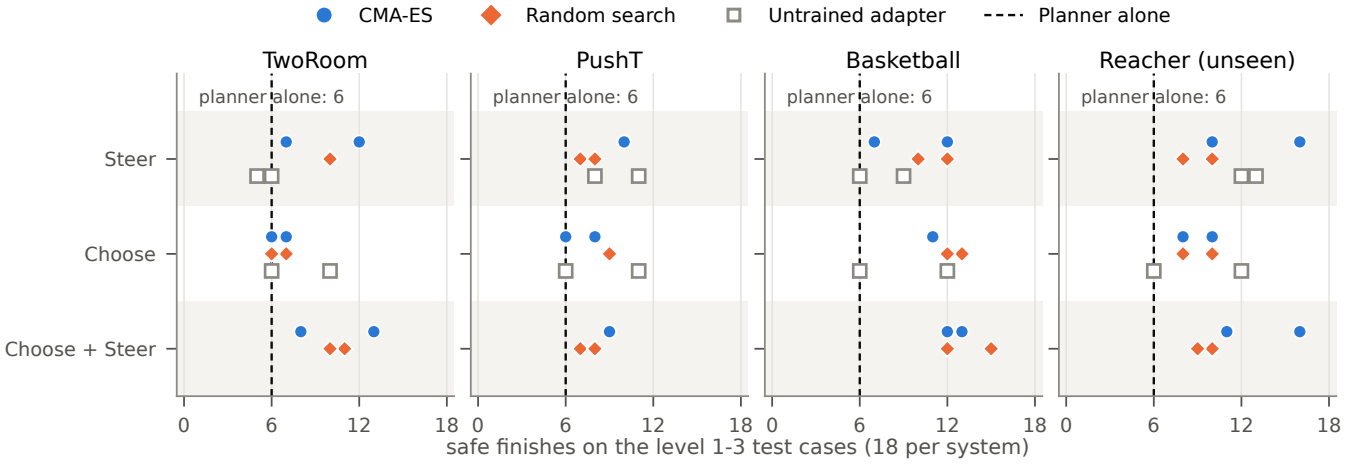


Figure 2: The dashed line is the planner alone. Nearly every selected adapter sits to its right, on every system. Random search (orange) reaches as far as CMA-ES (blue) on the training systems, but not on Reacher. Grey squares: untrained.

RQ2: a slight edge to CMA-ES, on the unseen system. Of 12 paired tests of CMA-ES against random search (each adapter type and repeat, on the training systems and on Reacher), only

Table 2: Safe finishes per level, of 6 cases each; bold cells beat the planner alone at that level, grey cells fall below it. On the goal (L4) has no safe finish, so it shows finishes / touches instead: fewer of both means the adapter stopped short.

	TwoRoom				PushT				Basketball				Reacher (unseen)			
Adapter	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
Planner alone	6	0	0	6/6	6	0	0	6/6	6	0	0	6/6	6	0	0	6/6
Steer, CMA-ES, repeat 1	3	**2**	**2**	3/3	6	**4**	0	2/6	6	**3**	**3**	0/4	6	**5**	**5**	0/5
Steer, CMA-ES, repeat 2	5	**4**	**3**	3/4	6	**3**	**1**	0/6	4	**2**	**1**	2/3	6	**4**	0	2/6
Steer, random search, repeat 1	5	**4**	**1**	3/3	5	**2**	0	0/5	4	**2**	**4**	3/5	6	**3**	**1**	2/5
Steer, random search, repeat 2	5	**2**	**3**	3/5	6	**2**	0	0/6	6	**3**	**3**	3/4	6	**2**	0	3/6
Steer, untrained adapter 1	4	**2**	0	6/6	5	**6**	0	4/6	6	**3**	0	4/4	6	**5**	**2**	1/6
Steer, untrained adapter 2	5	0	0	6/6	6	**2**	0	6/6	4	**2**	0	5/5	6	**4**	**2**	6/6
Choose, CMA-ES, repeat 1	6	**1**	0	1/1	6	**2**	0	4/6	5	**4**	**2**	5/5	6	**2**	0	6/6
Choose, CMA-ES, repeat 2	6	0	0	5/5	6	0	0	5/6	5	**4**	**2**	4/4	6	**3**	**1**	5/6
Choose, random search, repeat 1	6	**1**	0	6/6	6	**3**	0	4/6	6	**4**	**3**	5/6	6	**3**	**1**	6/6
Choose, random search, repeat 2	6	0	0	4/4	6	**3**	0	5/6	4	**4**	**4**	4/5	5	**2**	**1**	5/6
Choose, untrained adapter 1	6	**4**	0	2/2	6	**5**	0	4/6	6	**3**	**3**	3/6	6	**3**	**3**	6/6
Choose, untrained adapter 2	6	0	0	6/6	6	0	0	6/6	6	0	0	6/6	6	0	0	6/6
Choose + Steer, CMA-ES, repeat 1	3	**4**	**1**	1/1	6	**3**	0	0/6	6	**2**	**5**	0/1	6	**5**	**5**	0/6
Choose + Steer, CMA-ES, repeat 2	5	**5**	**3**	2/3	5	**3**	**1**	0/6	4	**5**	**3**	1/2	6	**4**	**1**	2/6
Choose + Steer, random search, repeat 1	5	**4**	**2**	1/1	6	**1**	0	0/0	4	**4**	**4**	2/5	6	**4**	0	1/4
Choose + Steer, random search, repeat 2	4	**4**	**2**	1/2	6	**2**	0	0/4	6	**4**	**5**	3/3	5	**3**	**1**	1/6

1 was significant: Steer, repeat 1, on Reacher, 6 cases to 0 ($p = 0.031$). Summed over the six pairs, CMA-ES won 24 cases and lost 8 on Reacher, but 38 and 43 on the training systems; Choose alone favoured random search (CMA-ES 10, random 17). The pairs share cases, so we do not test the sums. Most of the Reacher lead is one CMA-ES Steer adapter, counted in two pairs (Steer and Choose + Steer, repeat 1); where they were searched, the two tied.

RQ3: search made gains reliable, not larger. One untrained Choose adapter did as well as most selected adapters: 33 of 54, and 12 of 18 on Reacher, with 0 cases lost. The other opened its gate on only 33 of 96 test cases and matched the planner alone's safe finishes case for case: 18 and 6. The two untrained Steer adapters scored 26 and 19 of 54, each losing 3 cases. Every selected adapter beat the planner alone; only one beat the best untrained adapter.

On the goal: stopping short. No adapter can finish an On-the-goal case safely; what changes is how often it touches. The planner alone finishes and touches on all 24 cases. Steer and Choose + Steer often stop short instead: they touched on only 10 to 21 of 24. Choose stops less often; it touched on 18 to 24.

Training. Both methods beat the planner alone on the training cases within a few generations (Figure 3); CMA-ES's average sat higher on Choose but rose measurably in one search of four (Steer, repeat 1: Spearman $r_s = 0.25$, $p = 0.007$).

8 WHAT THE ADAPTERS DID

The gates opened on the hazards. On Graze and On-the-path cases (Figure 5), the gates of the selected adapters opened in nearly every episode: the CMA-ES Choose adapter of repeat 1, for example, in 48 of 48.

Steer moved the path outward (Figure 6). On TwoRoom the planner alone's mean closest approach to the contact ring is -0.91 steps (inside it); with Steer it ranged from -0.01 to 1.04. On basketball it moved from -0.62 to between 0.64 and 1.54. PushT moved least, from -0.81 to between -0.61 and -0.23.

Choose swapped plans only when its plan was too close. On the LeWM systems, when its gate opened, Choose's checker found a plan predicted to clear the hazard in 82 to 88% of decisions. It ran its best-keyed plan in 20 to 30% of open decisions; otherwise the planner's own plan was already clear enough, or no checked plan was clearer. It moved the closest approach less than Steer (TwoRoom -0.88 to -0.52 steps), fitting its higher finish rate.

9 BLOCK PLUCKING: AN ARM TASK

Block Plucking tests the same procedure on a task whose failures are its own, with no hazard added.

The task. A simulated Franka arm must pull the middle block out of a stack of three (Figure 7), in Isaac Sim 6.2 and Isaac Lab 3.0 at 20 Hz. We use the Hard variant, with a lighter top block and a smaller base. A failure is any of: a block falls below 0.13 m, the top block slides more than 0.07 m from the base, or the gripper leaves its boundary. A finish is *sustained* if it still holds 40 controls later with no failure, and here S counts sustained safe finishes.

The planner. It is AP again, with a memory of the 161,709 steps in the 1,369 successful demonstrations released with UNISafe [18]. Each decision retrieves the 64 most similar moments and considers the five recorded actions that followed each one. UNISafe's safety filter only watches: its signals are logged, never applied. On Hard probes it finished 5 of 12 but failed in 8 of 12, so finishing and staying safe pull against each other.

The adapter. A *segment selector* re-ranks the 64 retrieved moments inside the planner's final choice. Five weights w trade the planner's imagined cost z_i (Equation 1) against five standardized features y_i: retrieval distance, imagined cost, mean arm-action size, the share of gripper-closing commands, and progress through the demonstration. With $w = 0$ it is the planner alone:

$$i^\star = \arg\min_i \left(z_i - w^\top y_i\right). \tag{10}$$

A selector can only replay recorded behavior. We chose it after a screen of 24 random adapters of each kind on 4 starts: a residual on the planner's command finished 1 of 96 episodes and failed in 85, while the selector finished 76 and failed in 18.

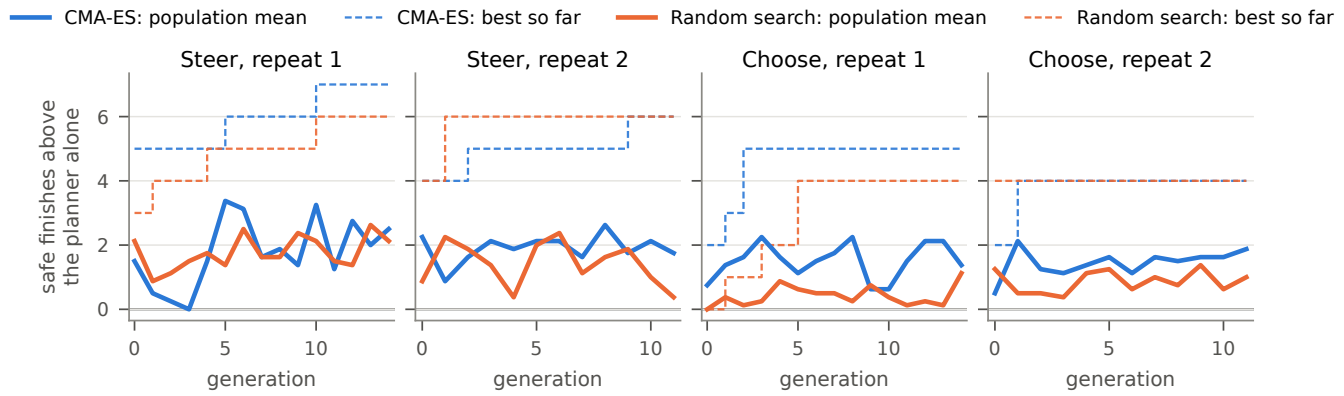


Figure 3: Solid lines are each generation's average and dashed steps the best found so far, both as safe finishes minus the planner alone's on the 12 training cases. Both methods beat the planner alone early, and neither's best pulls clearly ahead.

Table 3: Case-by-case comparisons on levels 1–3: cases gained–lost by the first method against the second, with the two-sided sign-test p, over the three training systems (54 cases) and on the unseen Reacher (18). Bold marks $p < 0.05$.

	CMA-ES vs planner alone				Random search vs planner alone				CMA-ES vs random search			
	Training		Reacher		Training		Reacher		Training		Reacher	
Adapter	g–l	p	g–l	p	g–l	p	g–l	p	g–l	p	g–l	p
Steer, repeat 1	14–3	**0.013**	10–0	**0.002**	13–4	**0.049**	4–0	0.12	9–7	0.80	6–0	**0.031**
Steer, repeat 2	14–3	**0.013**	4–0	0.12	13–1	**0.002**	2–0	0.50	9–10	1.00	2–0	0.50
Choose, repeat 1	9–1	**0.021**	2–0	0.50	11–0	**<0.001**	4–0	0.12	3–6	0.51	1–3	0.62
Choose, repeat 2	6–1	0.12	4–0	0.12	11–2	**0.022**	3–1	0.62	2–6	0.29	4–2	0.69
Choose + Steer, repeat 1	15–3	**0.008**	10–0	**0.002**	15–3	**0.008**	4–0	0.12	7–7	1.00	7–1	0.070
Choose + Steer, repeat 2	20–4	**0.002**	5–0	0.062	17–2	**<0.001**	4–1	0.38	8–7	1.00	4–2	0.69

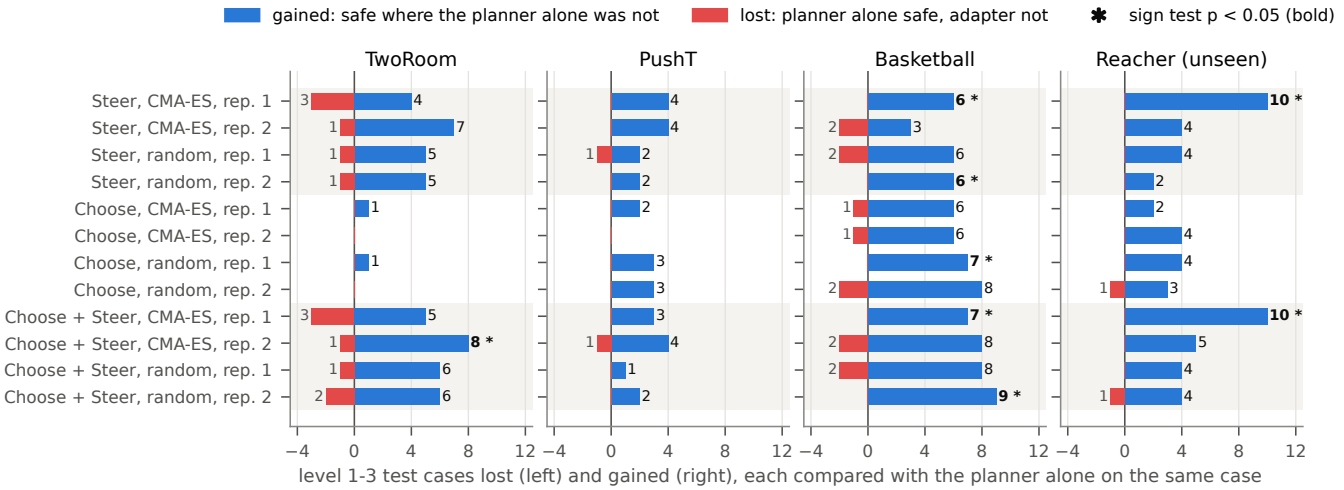


Figure 4: Blue bars are test cases gained over the planner alone, red bars cases lost, per selected adapter and system, with the counts printed on the bars. Gains outweigh losses in almost every row; bold with a star marks $p < 0.05$.

The search. Candidates are ranked by S, then finishes, then fewer failures. Diagonal CMA-ES (population 8, initial step size 0.5) ran 8 generations on two shared starts per generation. Random search ran 16 draws from the same starting distribution, $N(0, 0.5^2 I)$, on 8 starts, so each method used 128 episodes. The selected CMA-ES adapter is its final mean; the selected random one is its best draw. Each of three repeats was tested on 16 fresh starts. We fixed the rule first: over all 48 starts, the procedure *transfers* if CMA-ES gains at least 9 sustained safe finishes over the planner alone with no more failures, and shows *no signal* if both differences are within 3.

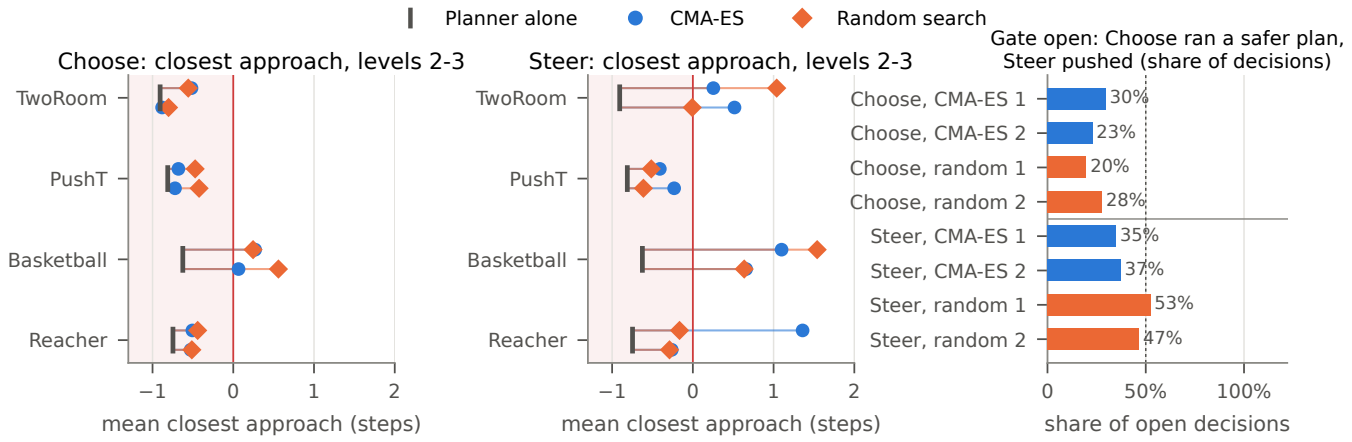


Figure 5: Left: the closest approach to the contact ring (below zero is a touch) for the planner alone and each selected adapter, on Graze and On-the-path cases. Right: what each adapter did when its gate was open (LeWM systems).

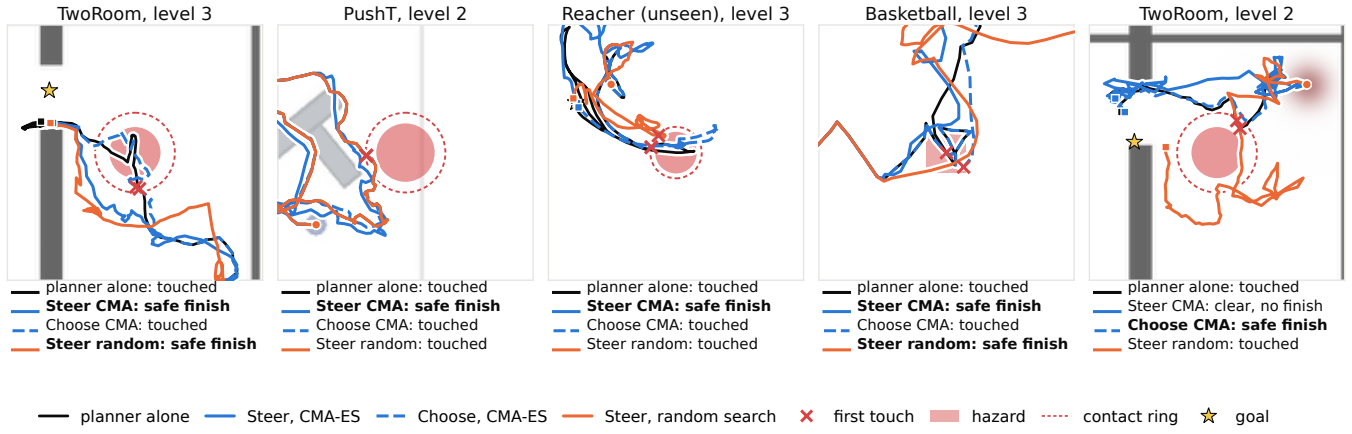


Figure 6: Recorded paths, one test case per panel (CMA-ES adapters from repeat 1, random search's from repeat 2); a red × marks the first touch. In the first four panels Steer from CMA-ES goes around the hazard and finishes safely; in the last it stays clear but stops short, and Choose finishes safely.

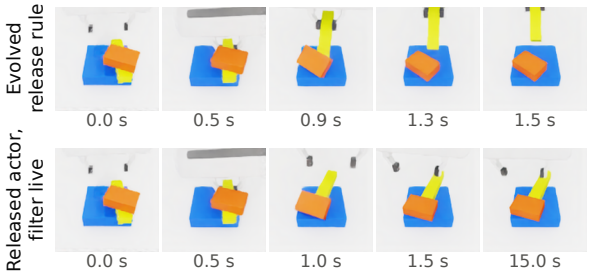


Figure 7: Block Plucking: the arm pulls the yellow middle block out of a stack of three without dropping or displacing the others. Frames from an earlier study on the regular task, not the pilot.

The planner alone is a coin toss. On 48 calibration starts on Hard it failed at a rate of 0.52 (Wilson 95% interval 0.38 to 0.66) and sustained a safe finish at only 0.29. It cannot see when the block is out, so it keeps following anchors.

The pilot found no signal (Figure 8). Over all 48 starts, CMA-ES and the planner alone both reached 10 sustained safe finishes, with 26 failures against 28. Start by start, CMA-ES gained 9 and lost 9 (McNemar exact $p = 1.00$). Three untrained adapters reached 12 and 25 failures; random search reached 7 and 30. All read no signal.

A residual looked like skill, but was not. We also tried a gated residual: a small correction to the arm joints, bounded at 0.1, that acts only when UNISafe's failure head predicts danger. With no search, 24 random residuals ran on the four starts where the planner alone failed. They finished safely in 48 of 96 episodes (ungated: 50 of 96). But a control bounded at 0.001, whose corrections are about 0.0004 per coordinate, did as well: 28 of 48 safe finishes. On a Hard start, any change to the actions re-rolls the episode from roughly even odds. The failure head also gave almost no warning: it passed 0.01 on only 2.5% of the planner alone's steps.

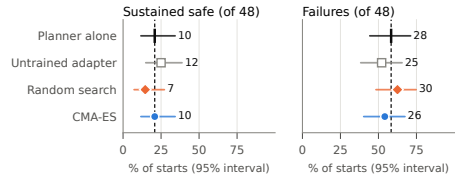


Figure 8: Block Plucking over 48 fresh starts: every interval overlaps the planner alone's, for sustained safe finishes (left) and failures (right).

The lesson. When an outcome turns on tiny differences, a gain counts as skill only if it beats a near-zero change. The graded systems lack such a control; there the evidence is the mechanism of Section 8: the closest approach moved outward.

10 DISCUSSION

The design did most of the work. Both adapters know which way is safe: Steer only pushes away, and Choose always prefers the clearer plan. With that built in, random genes are often good enough: one untrained adapter matched the selected ones, and random search matched CMA-ES on the systems it searched. Search made the gain reliable: every selected adapter beat the planner alone, while untrained adapters ranged from no change to as good as the best.

CMA-ES had little budget and little to learn. CMA-ES had 15 or fewer generations of 8 candidates on 12 cases. With little to learn beyond the built-in direction, random search can keep up.

Limits. Case sets are small (18 per system on levels 1–3, two repeats per method), so only large effects show, and we make no correction across tests. The adapters were redesigned after an earlier run on the same test cases, so those cases were not new to us. Hazards are placed against one recorded path, and the checker knows the hazard exactly, which a real robot may not. Nothing here certifies safety.

11 CONCLUSIONS

Can a small evolved adapter add safety to a frozen planner that cannot see the hazard? On hazards graded against the planner's own path, yes: adapters that know which way is safe raised safe finishes on the three training systems together, and the design transferred unchanged to an unseen system, where every selected adapter beat the planner alone. Safety cost some finishes. CMA-ES's adapters transferred slightly better than random search's, but one untrained adapter did nearly as well. On Block Plucking, search matched the planner alone, and random residuals did no better than a near-zero control. The next test of evolution needs adapters whose right behavior is not built in.

REFERENCES

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2017. Constrained Policy Optimization. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*. 22–31.

[2] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. 2021. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In *Advances in Neural Information Processing Systems*, Vol. 34.

[3] Eitan Altman. 1999. *Constrained Markov Decision Processes*. Chapman and Hall/CRC.

[4] Lenz Belzner and Martin Wirsing. 2021. Synthesizing Safe Policies under Probabilistic Constraints with Reinforcement Learning and Bayesian Model Checking. *Science of Computer Programming* 206 (2021), 102620.

[5] James Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research* 13 (2012), 281–305.

[6] Nicolas Carrara, Edouard Leurent, Romain Laroche, Tanguy Urvoy, Odalric-Ambrym Maillard, and Olivier Pietquin. 2019. Budgeted Reinforcement Learning in Continuous State Space. In *Advances in Neural Information Processing Systems*, Vol. 32.

[7] Weiqin Chen and Santiago Paternain. 2024. Generalized Constraint for Probabilistic Safe Reinforcement Learning. In *Proceedings of the 6th Annual Learning for Dynamics and Control Conference (Proceedings of Machine Learning Research, Vol. 242)*. 1606–1618.

[8] Kalyanmoy Deb. 2000. An Efficient Constraint Handling Method for Genetic Algorithms. *Computer Methods in Applied Mechanics and Engineering* 186, 2–4 (2000), 311–338.

[9] Nikolaus Hansen and Andreas Ostermeier. 2001. Completely Derandomized Self-Adaptation in Evolution Strategies. *Evolutionary Computation* 9, 2 (2001), 159–195.

[10] Nicklas Hansen, Hao Su, and Xiaolong Wang. 2024. TD-MPC2: scalable, robust world models for continuous control. In *International Conference on Learning Representations*.

[11] Kai-Chieh Hsu, Haimin Hu, and Jaime F. Fisac. 2023. The safety filter: a unified view of safety-critical control in autonomous systems. *Annual Review of Control, Robotics, and Autonomous Systems* (2023).

[12] Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. 2019. Residual reinforcement learning for robot control. In *IEEE International Conference on Robotics and Automation*.

[13] Xvyuan Liu, Jianjie Fang, Wei Wu, Chen Gao, and Yong Li. 2026. Aim Short to Reach Far: Your Frozen World Model Can Plan Better Than You Think. arXiv:2609.30036.

[14] Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. 2026. LeWorldModel: Stable End-to-End Joint-Embedding Predictive Architecture from Pixels. arXiv:2603.19312.

[15] Enrico Marchesini, Davide Corsi, and Alessandro Farinelli. 2022. Exploring Safer Behaviors for Deep Reinforcement Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 7701–7709.

[16] Raymond Ros and Nikolaus Hansen. 2008. A Simple Modification in CMA-ES Achieving Linear Time and Space Complexity. In *Parallel Problem Solving from Nature – PPSN X*. Springer, 296–305.

[17] Thomas P. Runarsson and Xin Yao. 2000. Stochastic Ranking for Constrained Evolutionary Optimization. *IEEE Transactions on Evolutionary Computation* 4, 3 (2000), 284–294.

[18] Junwon Seo, Kensuke Nakamura, and Andrea Bajcsy. 2025. Uncertainty-aware Latent Safety Filters for Avoiding Out-of-Distribution Failures. In *Proceedings of the 9th Conference on Robot Learning (Proceedings of Machine Learning Research, Vol. 305)*. 4442–4472.

[19] Tom Silver, Kelsey Allen, Josh Tenenbaum, and Leslie Kaelbling. 2018. Residual policy learning. *arXiv preprint arXiv:1812.06298* (2018).

[20] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. 2020. Meta-World: a benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*.