

Supplementary Information

Cognitive evolution enables de novo design of untethered millimeter-scale robots

Weizhi Zhao^{1,†}, Zhi Tao^{1,†}, Peihan Li¹, Xizhi Wang²,
Qinghong Wang², Boxian Wang², Yanxin Zhai¹, Wei Wei³,
Shitong Wang⁴, Shuangqi Li⁵, Tiantong Xu^{1,*}, Haiwang Li^{1,*}

¹Research Institute of Aero-Engine, Beihang University, Beijing 102206, China

²School of Energy and Power Engineering, Beihang University, Beijing 102206, China

³School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China

⁴School of Physics, Beihang University, Beijing 102206, China

⁵Department of Electrical and Electronic Engineering, The Hong Kong Polytechnic University,
Hung Hom, Kowloon, Hong Kong SAR, China

[†]These authors contributed equally.

*Correspondence to: xutiantong@buaa.edu.cn (T.X.) and lihaiwang@buaa.edu.cn (H.L.).

This Supplementary Information documents the design, evaluation and hardware validation of the untethered millimeter-scale robot reported in the main text. Supplementary Notes 1 and 2 define the actuation-system and locomotion-system design problems, their deterministic physical constraints and the finite-element and multibody-dynamics evaluators that supplied every objective value. Supplementary Notes 3 to 6 specify the optimizer configurations, the evaluation accounting and convergence rules applied identically to all methods, and the decision state, lineage memory and prompt interface of EvoAgent. Supplementary Notes 7 and 8 report the retrieval and lineage-memory ablations and the transfer and expert-feedback protocols. Supplementary Notes 9 to 11 describe actuator and robot fabrication, the hardware measurement setups and their statistics, and the literature comparator set used for the robot-level comparison. Supplementary Note 12 lists the movie legends.

Supplementary Note 1 Robot subsystem design problems and physical objectives

Actuation-system design problem

The actuation system of the robot is an EI-core electromagnetic linear actuator, and its design is formulated as a constrained physical black-box optimization problem. A candidate actuator is represented by

$$\mathbf{x}_{\text{act}} = (l_m, t_m, t_a, d_g, h_s, w_{\text{slot}}, h_{\text{slot}}, s, t_{b,\text{ratio}}), \quad (\text{S1})$$

where the slot-wall thickness, bridge height and winding-layer count used by the pre-simulation constraints are derived from geometry,

$$t_{\text{wall}} = \frac{h_s - h_{\text{slot}}}{2}, \quad t_b = t_{b,\text{ratio}} t_a, \quad n_2 = \lfloor 0.9 t_{\text{wall}} / w_{\text{coil}} \rfloor, \quad (\text{S2})$$

with $w_{\text{coil}} = 0.05 \text{ mm}$ and fixed arm width $w_a = 2.0 \text{ mm}$. These are the derived geometric quantities that enter the deterministic viability checks in Supplementary Table S2; additional post-simulation quantities such as volume, mass, force constant and actuator power metric are defined with the objective function below. The actuator variables were chosen to span the main magnetic-circuit, winding-space and package constraints of the EI-core linear electromagnetic actuator. Magnet length and thickness set the available remanent flux and effective magnetic-path length. Yoke thickness, bridge thickness and air-gap height jointly control magnetic reluctance, local saturation and air-gap field strength. Slot width and slot height determine the winding space and therefore the achievable turn count and force output, but excessive slot space reduces the iron cross-section and can increase saturation risk. These variables therefore span the central engineering tradeoff of the actuation system: raising air-gap field strength and force constant while preserving manufacturability within the volume and mass that an untethered millimeter-scale robot can carry. The search ranges used by all actuator optimizers are listed in Supplementary Table S1.

Table S1: Actuator design variables and search ranges.

Variable	Meaning	Unit	Range
l_m	Permanent-magnet length	mm	0.0–6.0
t_m	Magnet thickness	mm	0.3–0.5
t_a	Arm/yoke thickness	mm	0.35–0.65
d_g	Air-gap height	mm	0.3–0.65
h_s	Slot outer height	mm	1.2–2.2
w_{slot}	Slot width	mm	2.0–2.8
h_{slot}	Slot inner height	mm	0.8–1.3
s	Actuator stroke	mm	0.8–1.2
$t_{b,\text{ratio}}$	Bridge-to-yoke thickness ratio	–	1.5–2.0

Actuator constraints

All actuator proposals were filtered by the same deterministic pre-simulation constraints before Maxwell evaluation. Supplementary Table S2 lists the main actuator viability checks, their mathematical criteria and the physical meaning of each constraint.

Table S2: Actuator viability constraints used before Maxwell simulation.

Constraint	Criterion	Interpretation
Magnetic-gap clearance	$2d_g + t_b - h_s \geq 0.1 \text{ mm}$	Reserves vertical clearance between the air gap, bridge and slot envelope so that the magnetic gap and bridge are not geometrically collapsed.
Minimum stroke	$s \geq 1.0 \text{ mm}$	Requires the actuator stroke to be at least 1.0 mm, providing sufficient linear travel for robot-leg excitation.
Positive magnetic path	$l_m - s > 0$	Keeps the effective magnet-overlap length positive after accounting for the required stroke.
Slot-to-bridge clearance	$h_{\text{slot}} - t_b \geq 0.2 \text{ mm}$	Leaves manufacturable winding and insulation space above the bridge region.
Slot-width clearance	$w_{\text{slot}} - w_a \geq 0.2 \text{ mm}$	Keeps the coil slot wider than the fixed center arm, leaving lateral space for winding placement.
Air-gap wall clearance	$d_g - t_{\text{wall}} \geq 0.02 \text{ mm}$	Prevents the slot wall thickness from eliminating the designed air-gap clearance.
Overall slot width	$w_{\text{slot}} + 2t_{\text{wall}} < 4.0 \text{ mm}$	Keeps the slot-and-wall assembly within the actuator package width.
Overall actuator length	$l_m + 2t_a \leq 6.0 \text{ mm}$	Enforces the package-length limit of the millimeter-scale actuator.
Minimum winding layers	$n_2 \geq 3$	Ensures that the derived winding capacity is large enough to provide useful coil turns.

Actuator candidates first underwent deterministic pre-screening for geometry clearance, magnetic path, package size and winding-layer constraints. Feasible candidates proceeded to Maxwell finite-element evaluation; solver failures, invalid field extraction and severe local saturation received the predefined penalty.

Actuator objective function

For feasible actuators, volume V , mass m , force constant k_b and actuator power metric P_b were computed from the same Maxwell-derived field quantities and geometry. The scalar objective used for algorithmic comparison was

$$F_{\text{act}} = 0.5 \frac{V}{V_{\text{ref}}} + 0.5 \frac{m}{m_{\text{ref}}} - 4.0 \frac{k_b}{k_{b,\text{ref}}} - 1.0 \frac{P_b}{P_{b,\text{ref}}}. \quad (\text{S3})$$

Lower values indicate better actuator designs. The fixed engineering normalizers were $V_{\text{ref}} = 6.7 \times 10^{-8} \text{ m}^3$, $m_{\text{ref}} = 3.79 \times 10^{-4} \text{ kg}$, $k_{b,\text{ref}} = 0.239 \text{ N A}^{-1}$ and $P_{b,\text{ref}} = 0.819$ in the same units as P_b .

The weights in Eq. S3 reflect the robot-level requirements and were fixed before optimization. Equal coefficients of 0.5 were assigned to the normalized volume and mass terms so that packaging space and carried mass were penalized jointly, without either term dominating the search. The force-constant coefficient of 4.0 was selected as the primary performance term because thrust per unit current determines low-current operation and payload margin of the assembled robot. The dynamic-output coefficient of 1.0 provided a secondary preference for useful stroke and mover dynamics while retaining this priority ordering. The actuator power metric

$$P_b = 4k_b^{1.5} s^{0.5} m_a^{-0.5} \quad (\text{S4})$$

is a relative dynamic-output metric rather than an electrical input power. Here s is the stroke coordinate and m_a is the mover mass. The form follows the scaling $P \sim Fv$, with $F \propto k_b I$ and a characteristic velocity scaling as $(Fs/m_a)^{1/2}$; after removing the common current factor, the remaining design-dependent term increases with force coefficient and stroke and decreases with mover mass. The score therefore rewards actuators that deliver high current-to-force gain and useful dynamic stroke while remaining small and light, which are the properties that decide whether an actuator can drive a robot from onboard power. The four weights were held fixed for all methods.

Locomotion-system design problem

The locomotion system converts actuator stroke into forward motion of the robot body. Its design used a five-dimensional continuous genotype,

$$\mathbf{x}_{\text{leg}} = (m, n, \alpha, \beta, d_{\text{bat}}), \quad (\text{S5})$$

where m is front-leg length, n is rear-leg length, α is rear-foot ground angle, β is front-foot ground angle and d_{bat} is the battery center offset. The front- and rear-leg lengths and foot angles determine the reconstructed body posture, foot-contact locations and leg-linkage geometry. Because the robot carries its own battery, the battery-center offset sets the load distribution between the front and rear feet and therefore the contact state on which forward motion depends. Body height, foot span, joint angle and geometric feasibility are derived during deterministic reconstruction and viability screening. The score is the net forward displacement of the robot center of mass over a one-second multibody-dynamics simulation, which is the locomotion quantity that the assembled robot must ultimately deliver.

Table S3: Robot-leg design variables and search ranges.

Variable	Meaning	Unit	Range
m	Front-leg length	mm	0.0–6.0
n	Rear-leg length	mm	0.0–5.0
α	Rear-foot angle	degree	80–130
β	Front-foot angle	degree	50–120
d_{bat}	Battery center offset	mm	0.0–9.0

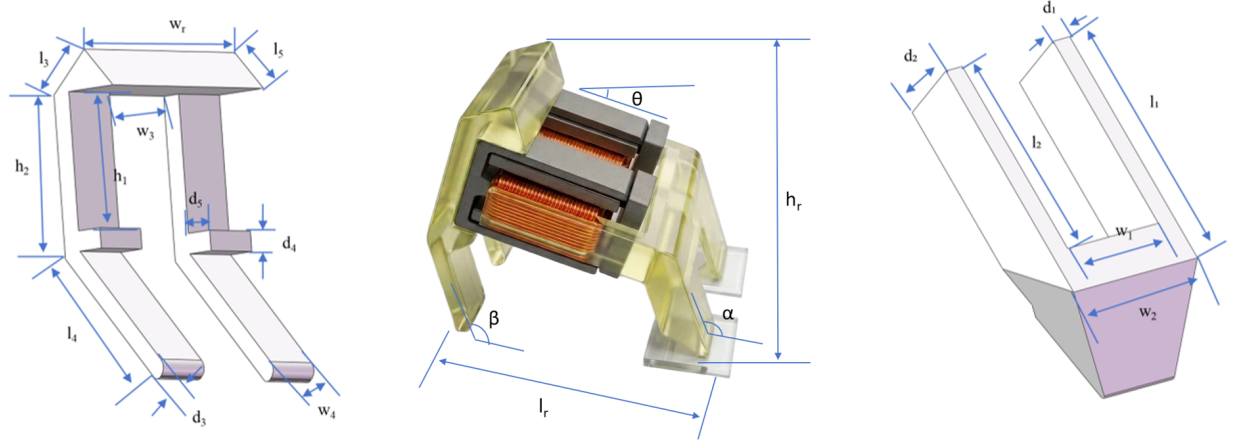


Figure S1: Geometric definitions used by the robot-leg validator. The rear foot contact point is denoted D , the rear knee point G , the rear body-joint point A , the front body-joint point B , the body reference point C , and the front foot contact point F .

Point and quantity definitions. The geometric symbols used by the robot-leg validator are summarized in Supplementary Table S4. These definitions are used consistently in the reconstruction equations and viability filters below.

Table S4: Geometric points and derived quantities used by the robot-leg validator.

Symbol	Definition
D	Rear foot contact point fixed at the ground origin, $D = (0, 0)$.
G	Rear knee point reconstructed from rear-leg length n and rear-foot angle α .
A	Rear body-joint point obtained by offsetting G by the fixed rear-body geometry.
B	Front body-joint point reconstructed from the rigid body length L_{AB} and front-leg height.
C	Body reference point; its vertical coordinate C_y is the body-ground clearance used by the validator.
F	Front foot contact point on the ground, reconstructed from B , front-leg length m and front-foot angle β .
DF	Horizontal distance between rear and front foot contacts, $DF = F_x $.
$H_{\max}$	Maximum reconstructed body height after adding the assembly offset.
D_{12}	Front-leg joint angle used to reject over-extended front-leg configurations.

Robot-leg geometric reconstruction

All leg constraints were evaluated by a deterministic planar geometry reconstruction before Adams simulation. Length variables were expressed in millimeters as

$$m_{\text{mm}} = m, \quad n_{\text{mm}} = n. \quad (\text{S6})$$

The rear foot contact point was fixed at

$$D = (0, 0). \quad (\text{S7})$$

Given rear-leg length n_{mm} and rear-foot angle α , the rear knee point was

$$G = (-n_{\text{mm}} \cos \alpha, n_{\text{mm}} \sin \alpha). \quad (\text{S8})$$

The rear body-joint point A was offset from G by the fixed body geometry parameter $AG = 2.20$ mm,

$$A = (A_x, A_y) = G + (2.20 \sin \alpha, 2.20 \cos \alpha). \quad (\text{S9})$$

The front foot contact point lies on the ground, so its vertical coordinate is zero. The front leg gives

$$B_y = m_{\text{mm}} \sin \beta. \quad (\text{S10})$$

The rigid-body distance between the rear and front body-joint points is

$$L_{AB} = \sqrt{BC^2 + AC^2}, \quad BC = 7.26 \text{ mm}, \quad AC = 1.95 \text{ mm}. \quad (\text{S11})$$

The stride-span feasibility condition is therefore

$$|B_y - A_y| \leq L_{AB}. \quad (\text{S12})$$

When this holds, the horizontal separation between A and B is reconstructed as

$$\Delta x = \sqrt{L_{AB}^2 - (B_y - A_y)^2}, \quad \theta_{AB} = \arctan 2(B_y - A_y, \Delta x). \quad (\text{S13})$$

The body pitch angle is obtained by subtracting the fixed body-frame angle

$$\theta_{\text{body}} = \theta_{AB} - \arctan 2(-AC, BC). \quad (\text{S14})$$

The body reference point C and the front body-joint point B are then

$$C = (C_x, C_y) = A + (AC \sin \theta_{\text{body}}, -AC \cos \theta_{\text{body}}), \quad (\text{S15})$$

$$B = (B_x, B_y) = C + (BC \cos \theta_{\text{body}}, BC \sin \theta_{\text{body}}). \quad (\text{S16})$$

Thus C_y is the vertical height of the body reference point above the ground. The front foot contact point is

$$F = (F_x, 0) = (B_x + m_{\text{mm}} \cos \beta, 0), \quad (\text{S17})$$

and the foot-span constraint uses

$$DF = |F_x|. \quad (\text{S18})$$

Robot-leg viability constraints

The robot-leg validator enforced the seven constraints in Supplementary Table S5. These deterministic checks were applied before Adams simulation for EvoAgent and all robot-leg baselines. The height criterion used

$$H_{\text{max}} = \max(C_y, Y_{BR}, Y_{HT}, Y_{TR}) + 5.8 \text{ mm}, \quad (\text{S19})$$

and Ω_{body} denotes the rigid-body interior after transformation into the body frame.

Table S5: Robot-leg viability constraints used before Adams simulation.

Constraint	Criterion	Interpretation
Stride-span feasibility	$ B_y - A_y \leq L_{AB}$	Ensures that the reconstructed front and rear body-joint heights can be connected by the rigid body length.
Body-ground clearance	$C_y > 0$	Keeps the body reference point above the ground plane after geometric reconstruction.
Rear-leg anti-folding	$A_y \cos \theta_{\text{body}} - A_x \sin \theta_{\text{body}} > 0$	Prevents rear-leg configurations that fold backwards relative to the body frame.
Body/foot penetration	$G, D, F \notin \Omega_{\text{body}}$	Rejects geometries in which leg or foot points penetrate the rigid body envelope.
Maximum body height	$H_{\text{max}} \leq 15 \text{ mm}$	Limits the assembled robot height.
Front-leg joint angle	$D_{12} \leq 90^\circ$	Enforces a front-leg joint angle compatible with 3D-printing and assembly.
Foot-span limit	$DF \leq 13 \text{ mm}$	Limits the horizontal distance between rear and front foot contacts to keep the robot footprint within the intended body scale.

Candidates that passed pre-screening but produced solver failures, invalid outputs or dynamics failures received the predefined outcome: actuator failures received the penalty value, whereas locomotion outcomes such as robot-flight, in which the simulated robot loses ground contact, or robot-fall received zero forward displacement. EvoAgent stored these outcomes in the lineage record after the evaluation round, so that a failure of the robot to move remained available as physical evidence for later proposals.

Supplementary Note 2 Physical evaluators for the actuation and locomotion systems

Maxwell actuator evaluation. Actuator candidates were evaluated through an automated ANSYS Electronics Desktop/Maxwell 3D magnetostatic finite-element pipeline (ANSYS Electronics Desktop 2023 R1, PyAEDT automation, millimeter model units). For each candidate, the wrapper generated geometry, assigned materials, meshed the model, solved the field and post-processed average air-gap flux density and regional saturation. The two NdFeB magnets were assigned opposite coercivity directions; iron-core components and the surrounding region used the same assignments across all methods.

The Maxwell model used an external air region with 40% padding in all directions. The region-level length operation imposed a maximum element length of 1 mm. A second operation enabled internal adaptive refinement over the iron core, magnets and surrounding region; the adaptive solve used 30% refinement per pass with a 1% error target. The magnetostatic setup used maximum passes = 10, minimum passes = 2 and minimum converged passes = 1. The iterative solver option was disabled. The workflow explicitly generated the mesh when possible, saved the project, ran **AnalyzeAllNominal**, and saved the project again after solution.

Post-processing used FieldsReporter. The average air-gap B_z was extracted from the observation

volume Box1. Mean $|B|$ values for the upper-yoke and middle-block saturation checks were extracted from two planes, BsatPlane_ta and BsatPlane_tb. The actuator force constant used in the score was then computed consistently for all methods as

$$k_b = 2n_{\text{turns}}\bar{B}w_a, \quad n_{\text{turns}} = n_1n_2. \quad (\text{S20})$$

SolidWorks parametric robot modeling. Robot-leg candidates were first written to `input.csv`. A preprocessing script converted the optimization variables into the derived geometric parameters required by the CAD model and wrote them to `params.csv`. SolidWorks was controlled through COM automation. The workflow connected to, or launched, a SolidWorks session, executed the parameter-editing macro `sw_edit.swp`, executed the assembly macro `sw_asm.swp`, and exported the updated robot geometry as the Parasolid file `mircorob.x_t`. The same parameter-conversion, macro and export pipeline was used for EvoAgent and robot-leg baselines.

Adams robot-leg evaluation. The Parasolid geometry was imported into MSC Adams 2024 through `mdi.bat` and the command file `adams.cmd`. The Adams model used SI units (meter, degree, newton, kilogram and second). Rigid-body densities were set to 5800 kg m^{-3} for the core and magnet, 100 kg m^{-3} for the front leg, 1100 kg m^{-3} for the rear leg and $1.6345 \times 10^4 \text{ kg m}^{-3}$ for the battery. The front leg, core, magnet and battery were connected through fixed joints as specified in the command file.

Table S6: Adams contact and force settings used for robot-leg simulation.

Model component	Setting	Value
Backleg-magnet contacts	Solid-to-solid, no Coulomb friction	stiffness 10^8 , damping 10^4 , exponent 2.2, $d_{\text{max}} = 10^{-4}$
Backleg-core contacts	Solid-to-solid, no Coulomb friction	stiffness 10^8 , damping 10^4 , exponent 2.2, $d_{\text{max}} = 10^{-4}$
Backleg-ground contacts	Coulomb friction enabled	$\mu_s = 2.0$, $\mu_d = 2.0$, stiction transition velocity 0.1, friction transition velocity 1.0
Frontleg-ground contact	Coulomb friction enabled	$\mu_s = 0.3$, $\mu_d = 0.1$, stiction transition velocity 0.1, friction transition velocity 1.0
Gravity	Body gravitational field	$g_y = -9.80665 \text{ m s}^{-2}$
Actuation force	Pair of equal and opposite single-component forces	$\pm 1500 \times 0.001 \times 13.5 \times 10^{-3} \text{ IF}(\sin(20 \cdot 2\pi t) : 1, 0, -1) \text{ N}$
Simulation	Transient single run	end time 1.0 s, 2000 steps

The result parser extracted the battery-center displacement in x , y and z , the vertical displacement range, total three-dimensional displacement, average velocity in x and total average velocity. The optimization objective used net forward displacement in millimeters. Failed evaluations were archived with input parameters, logs and final result files when available. Outcomes with excessive vertical displacement were categorized as robot-flight, robot-fall or exceeded vertical range according to the same thresholds for all robot-leg methods.

Supplementary Note 3 Optimizer configurations, convergence criteria and accounting

Common evaluation accounting and convergence rule. All methods in both design problems followed one accounting and convergence protocol. Deterministic pre-screening rejections were recorded as proposals outside the simulation-evaluation budget. Each simulator run that terminated normally and returned either a numerical objective value or a predefined, physically interpretable failure outcome was recorded as a successful simulator evaluation. Predefined physical outcomes were retained in this record; for example, a robot-flight outcome returned zero forward displacement and therefore counted as a successful simulator evaluation. Starting from the first successful simulator evaluation, convergence was assessed with a trailing 20-evaluation window. Let B_t denote the cumulative-best objective after t successful simulator evaluations, and let $\epsilon > 0$ be a numerical safeguard. For the actuator minimization task,

$$I_t^{\min} = \frac{B_{t-20} - B_t}{\max(|B_{t-20}|, \epsilon)}, \quad (\text{S21})$$

whereas for the locomotion-system maximization problem,

$$I_t^{\max} = \frac{B_t - B_{t-20}}{\max(|B_{t-20}|, \epsilon)}. \quad (\text{S22})$$

A run reached the convergence window when $I_t \leq 0.01$. The resulting number of successful simulator evaluations to convergence was used for Fig. 2b, Fig. 2d and Supplementary Tables S8 and S9. Lower actuator scores and higher robot-leg forward-displacement scores indicate better performance.

The baseline set covered common candidate-generation and update mechanisms used in expensive black-box optimization. GA and PSO represent population-evolution and swarm-search methods, and the GA condition is reported as the Darwinian baseline in the main text; BO represents Gaussian-process surrogate optimization with an acquisition function; SMAC and BORE represent non-Gaussian surrogate or classifier-based optimization strategies; PFN-CEI and POM represent pre-trained or learned optimizers; and LLM-only isolates candidate generation by the language model without external retrieval or lineage memory. EvoAgent adds retrieval, lineage memory, pre-evaluation repair and critic review to this proposal stage. All methods used the same variable ranges, deterministic constraints, physical evaluators, objectives and failure-accounting rules. The actuation-system problem used the Maxwell automation wrapper and post-processing metrics described above; the locomotion-system problem used geometric reconstruction, SolidWorks parameterization and Adams dynamics evaluation.

Table S7: Optimizer configurations for the actuation-system design problem. GA is the condition reported as the Darwinian baseline in the main text.

Method	Proposal/update family	Stage prior	Proposal revision	Key configuration
GA	Population mutation and selection	no	no	Population 20, crossover 0.8, mutation 0.1, elite size 2
PSO	Swarm update	no	no	Swarm 20, inertia $w = 0.7$, $c_1 = c_2 = 1.5$
BO	Gaussian-process surrogate	no	no	10 initial samples, expected-improvement acquisition
SMAC	Random-forest surrogate	no	no	20 initial samples, 10 trees
BORE	Density-ratio surrogate	no	no	$\gamma = 0.25$, 64-32 classifier, 100 epochs
PFN-CEI	Pretrained surrogate acquisition	no	no	10 initial samples, 5000 candidates per round
POM	Pretrained optimizer	no	no	Zero-shot optimization model
LLM-only	Language-model mutation	no	prompt repair only	Same backend model; no retrieval or lineage memory
EvoAgent	Cognitive mutation	yes	yes	Diagnosis, retrieval, pre-screening and critic ensemble
EvoAgent (FB)	Feedback-assisted cognitive mutation	yes	yes, with expert directive	EvoAgent plus lineage-level expert feedback

Table S8: Actuation-system optimization results across three independent runs. GA is the Darwinian baseline of the main text. CI denotes a Student’s- t 95% confidence interval. N_{conv} is the mean number of successful simulator evaluations to convergence.

Method	Mean cumulative-best score	95% CI	Best run	$\bar{N}_{\text{conv}}$
GA	−9.71	±3.57	−11.03	55.3
PSO	−10.24	±1.02	−10.70	87.3
BO	−10.65	±4.71	−11.85	38.7
SMAC	−9.36	±3.32	−10.59	57.7
BORE	−9.08	±0.67	−9.30	45.0
PFN-CEI	−11.45	±0.27	−11.51	51.0
POM	−9.95	±0.00	−9.95	57.0
LLM-only	−12.69	±8.10	−16.37	62.7
EvoAgent	−17.38	±0.25	−17.46	42.7
EvoAgent (FB)	−17.08	±0.26	−17.17	33.3

Table S9: Locomotion-system optimization results across three independent runs. GA is the Darwinian baseline of the main text. CI denotes a Student’s- t 95% confidence interval. N_{conv} is the mean number of successful simulator evaluations to convergence.

Method	Mean cumulative-best score	95% CI	Best run	$\bar{N}_{\text{conv}}$
GA	17.43	± 14.97	21.86	45.3
PSO	19.90	± 5.49	22.01	55.3
BO	18.16	± 5.22	20.35	53.0
SMAC	19.25	± 4.59	21.24	53.0
BORE	17.11	± 7.79	20.23	38.7
PFN-CEI	21.62	± 2.75	22.89	55.7
POM	21.83	± 4.27	23.81	47.3
LLM-only	24.03	± 3.08	25.20	46.0
EvoAgent	25.27	± 1.04	25.72	39.7
EvoAgent (FB)	25.68	± 0.82	26.05	30.7

Supplementary Table S10 reports proposal-budget accounting for actuator optimization runs with archived proposal traces. Generated proposals are candidates emitted by the optimizer; validated descendants passed deterministic pre-screening; successful simulator evaluations returned a valid Maxwell score.

Table S10: Proposal-budget accounting for fully instrumented actuator optimization runs.

Method / audit run	Generated	Rejected	Validated	Successful simulator evaluations	Improved
GA audit	1155	952	200	200	13
LLM-only audit	108	47	61	61	13
EvoAgent audit	102	56	44	44	20

Supplementary Note 4 EvoAgent proposal state and pre-evaluation decision layer

This note describes the decision layer that acts before each expensive physical evaluation of a candidate robot subsystem. The large language model proposed candidate edits from past lineage records, retrieved priors and constraint feedback. Deterministic validators and physical simulators supplied the accepted objective values.

Sequential decision state. At generation t , EvoAgent receives the current design, incumbent best design, recent lineage records, retrieved knowledge, accumulated experience and constraint feedback,

$$s_t = (\mathbf{x}_t, \mathbf{x}_t^*, F_t^*, \mathcal{H}_t, \mathcal{K}_t, \mathcal{E}_t, \mathcal{C}_t). \quad (\text{S23})$$

Here $\mathbf{x}_t$ is the current genotype, $\mathbf{x}_t^*$ and F_t^* are the incumbent best design and score, $\mathcal{H}_t$ is the recent trajectory, $\mathcal{K}_t$ is retrieved knowledge, $\mathcal{E}_t$ is the accumulated lineage memory and $\mathcal{C}_t$ stores critic and constraint feedback. Proposal generation uses the information available at round t , followed by deterministic screening and physical evaluation.

Retrieval-augmented engineering priors. The retrieval module supplied domain priors before each expensive evaluation. The corpus contained electromagnetic actuator references, winding and miniaturization guidelines, material information and accumulated lineage records. The default index recursively ingested `.txt`, `.md`, `.log`, `.docx` and `.pdf` files. Documents were divided into 900-character chunks with 150-character overlap; chunks shorter than 120 characters were excluded. Chunks were embedded using `openai/text-embedding-3-small` through an OpenAI-compatible OpenRouter endpoint. In the reported optimization path, the task-level system prompt was the embedding query, while the current design state, active constraints, saturation status and recent failure modes entered through lineage and strategy context blocks. The four highest-scoring cosine-similarity passages were supplied to the agent as an 800-character retrieval context. The retrieved knowledge $\mathcal{K}_t$ grounded the next mutation.

Proposal format. Each candidate generated by the language model contained three fields: (i) proposed parameter edits, (ii) physical rationale and expected performance trends, and (iii) risk notes identifying likely constraint activations. The parser extracted the structured parameter values and discarded unparseable free text. Values outside the global bounds were clipped and passed to the deterministic constraint validator. Candidates that failed parsing or validation were revised when allowed by the configuration, or recorded as discarded generated proposals.

Table S11: Per-round information flow in the EvoAgent decision layer.

Stage	Main inputs	Archived output
Context assembly	Incumbent design, recent lineage, retrieved priors, active constraints	Structured prompt context and retrieval summary
Candidate generation	Context state s_t and task-specific variable bounds	Proposed genotype, rationale and risk notes
Parsing and repair	Structured proposal text and global bounds	Parsed genotype or discarded proposal record
Pre-simulation screening	Parsed genotype and deterministic constraints	Accepted candidate, repaired candidate or infeasible proposal
Physical evaluation	Maxwell or Adams wrapper output	Objective value, diagnostic fields and evaluator status
Lineage update	Outcome, critic prediction and constraint feedback	Instance record, reflection trigger and updated strategy notes

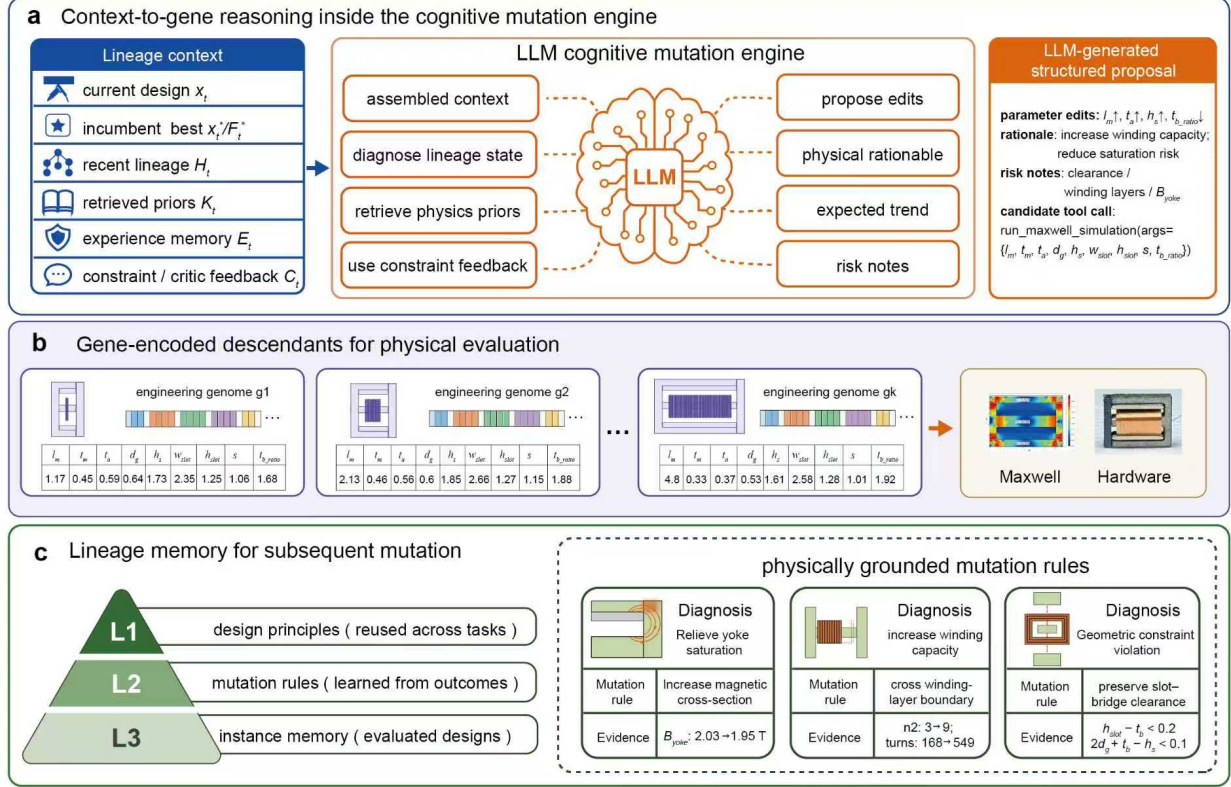


Figure S2: Representative evidence for cognitive mutation and lineage inheritance. **a**, Context-to-gene reasoning inside the cognitive mutation engine. The current design, incumbent-best design, recent lineage, retrieved physical priors, accumulated experience and constraint or critic feedback form the lineage context. The large language model diagnoses this context and returns a structured proposal containing parameter edits, physical rationale, expected performance trends, risk notes and a candidate simulation call. **b**, Gene-encoded actuator descendants submitted to physical evaluation. Parsed genomes are screened by deterministic constraints, rendered as electromagnetic designs and evaluated in Maxwell; the selected design subsequently proceeds to hardware validation. **c**, Lineage memory for subsequent mutation. Evaluated designs and their diagnoses are retained as instance records, while recurring outcomes are compressed into physically grounded mutation rules and higher-level design principles. Representative records show how diagnoses of yoke saturation, winding capacity and geometric constraint violation are converted into reusable mutation rules.

Outcome-based reward and lineage update. The reward signal used for lineage logging and strategy update was computed by the optimization harness rather than by changing the physical objective in Eq. S3. For actuator runs, a valid Maxwell result contributed a performance term

$$r_{\text{perf}} = 5 \frac{F_{\text{ref}} - F_{\text{act}}}{|F_{\text{ref}}|} + 0.5 \min\left(\frac{k_b}{0.3}, 2\right) + 0.3 \min\left(\frac{P_b}{1.0}, 2\right), \quad (\text{S24})$$

with $F_{\text{ref}} = -10.0$. If a valid result improved over the incumbent, an additional improvement bonus was applied,

$$r_{\text{imp}} = 5.0 \min\left(10 \frac{F_{\text{best}} - F_{\text{act}}}{|F_{\text{best}}|}, 2\right), \quad (\text{S25})$$

where F_{best} is the previous incumbent score. Deterministic constraint violations received a reward of -10.0 and simulation failures received -20.0 ; a constraint-violating case that still returned a field quantity received a $+2.0$ partial diagnostic credit. Exploration credit used a novelty threshold of 0.2 in the normalized actuator-parameter space and a maximum exploration bonus of 2.0 . Expert-feedback consistency, when used, contributed $+2.0$ for compliant edits or -3.0 otherwise. These quantities were recorded in the lineage logs.

Dense state-value and critic signals. For the actor-critic diagnostic layer, the state-value estimate used $k = 5$ nearest neighbors in normalized actuator-parameter space. Distances were computed after normalizing each parameter by its search range, and neighbor weights used a Gaussian kernel,

$$w_i = \exp\left[-\frac{d_i^2}{2(0.3)^2}\right], \quad V(s_t) = \frac{\sum_i w_i v_i}{\sum_i w_i}, \quad (\text{S26})$$

where v_i is the value transformed from the archived score. The temporal-difference diagnostic used $\gamma = 0.95$,

$$\delta_t = r_t + 0.95V(s_{t+1}) - V(s_t). \quad (\text{S27})$$

A four-member critic ensemble reviewed each proposal before simulation. The critics checked magnetic-circuit plausibility, consistency with recent performance trends, proximity to deterministic constraint boundaries and proposed step size. Each critic returned a score $q_m \in [-1, 1]$ and confidence $c_m \in [0, 1]$. The default critic weights were 0.35 for the magnetic critic, 0.30 for the performance critic, 0.20 for the constraint critic and 0.15 for the magnitude critic, giving

$$r_{\text{dense}} = \frac{\sum_m a_m c_m q_m}{\sum_m a_m c_m}. \quad (\text{S28})$$

When dense critic feedback was available, the harness combined it with the real outcome reward after soft normalization,

$$r_t = 0.6 \tanh(r_{\text{real}}/5.0) + 0.4r_{\text{dense}}, \quad (\text{S29})$$

and clipped the result to $[-1, 1]$. The critic signal was used for feedback and lineage-memory updates.

Table S12: Critic ensemble used before expensive evaluation.

Critic	Input evidence	Main rejection mode
Magnetic-circuit critic	Flux path, air gap, yoke/bridge dimensions, saturation history	High saturation risk or broken magnetic path
Performance-trend critic	Recent score changes and parameter sensitivities	Proposed direction contradicts observed trend
Constraint-boundary critic	Deterministic constraint margins	Candidate too close to infeasible boundary
Magnitude critic	Normalized parameter step size	Step too small to be informative or too large to remain controlled

Supplementary Note 5 Lineage memory, reflection and transfer rules

This note records how design outcomes were stored and reused across generations. The lineage-memory system retained append-only records of accepted candidates, failed proposals and evaluator diagnostics. Each lineage record included proposal rationale, expected performance trends, risk notes, constraint margins, simulation diagnostics, critic judgments, failure category and reflection, as summarized in Supplementary Table S13.

Lineage-memory hierarchy. EvoAgent maintained structured lineage memory beyond a score history. The hierarchy comprised instance memory (L3), mutation rules (L2) and design principles (L1), as shown in Supplementary Fig. S2. Instance memory stored generated proposals, rationales, constraints, evaluator outcomes and post-hoc diagnoses. Recurring physical relations, including clearance, saturation and winding-layer transitions, were compressed into mutation rules. Repeatedly supported rules were further abstracted into design principles that could be reused across related tasks.

Short-term and long-term lineage memory. Short-term lineage memory retained the recent lineage window used for immediate diagnosis. Long-term lineage memory persisted across the run as an append-only stream of observations, failed proposals, accepted descendants, field diagnostics, expert feedback and reflections. Each record was tagged with type, round index, score, constraint status, importance and transfer scope.

Table S13: Fields stored in each lineage-memory record.

Field group	Stored content
Design state	Round index, genotype, normalized genotype, parent identifier and task bounds
Proposal rationale	Model-provided physical rationale, expected performance trends and risk notes
Validation result	Parser status, deterministic constraint margins, repair status and rejection reason when applicable
Evaluator output	Physical score, raw diagnostic metrics, solver status and failure category
Learning signal	Incumbent improvement, dense-value estimate, critic prediction error and reflection trigger
Reuse tags	Importance score, transfer scope, related rule identifiers and whether the record entered prompt context

Reflection and rule extraction. Reflection was triggered by stagnation, repeated infeasibility, large degradation, saturation warnings or periodic checkpoints. Each reflection used the form: what happened, why it happened, what was learned and what action should be preferred or avoided next. When repeated observations supported the same relation, the agent promoted the reflection into a reusable mutation rule. Examples include slot-height/winding-layer coupling, saturation-driven yoke thickening and air-gap reduction under clearance safety.

Table S14: Reflection triggers and resulting lineage-memory actions.

Trigger	Diagnostic question	Memory action
Stagnation	Which parameter directions have stopped improving the incumbent?	Store an exploitation-limit note and recommend a new local direction or discrete transition.
Repeated infeasibility	Which deterministic constraint is repeatedly active?	Promote the constraint margin into a rule used before proposing the next candidate.
Large degradation	Which edit most plausibly caused the score drop?	Mark the edit as high risk and reduce its priority in future proposals.
Saturation warning	Which yoke, bridge or gap region is responsible?	Store a magnetic-path rule linking geometry edits to saturation relief.
Expert feedback	What high-level directive was supplied and when?	Store the directive as a lineage-level strategy note. Candidate parameter values remain generated by the optimizer.

Human feedback. Expert feedback was stored as a lineage-level directive. Feedback could highlight a likely discrete transition, warn against saturated subspaces or prioritize miniaturization. EvoAgent then generated a concrete candidate, passed it through the same deterministic constraints and submitted it to the same physical evaluator.

Transfer mechanisms. For transfer experiments, variables were normalized to the unit interval using the target-task bounds. Raw transfer reused archived instance and rule records directly as context. Distilled transfer used compact natural-language summaries of physical design principles. Hybrid transfer combined both. The transferred object was the structured lineage-memory record.

Table S15: Three-level knowledge hierarchy for experience transfer.

Level	Stored content	Transfer scope
L1 design principles	Physics-grounded abstract rules such as reducing air gap under clearance safety or increasing magnetic cross-section under saturation	Cross-topology and cross-domain transfer
L2 mutation rules	Sensitivity rankings, constraint-boundary rules, parameter couplings	Same topology or parameter-regime shift
L3 instance memory	Parameter values, evaluator output, constraint status, rationale	Same-task warm start

Supplementary Note 6 EvoAgent prompts and implementation disclosure

This note records the run-level prompts and agent-interface fields used to make EvoAgent proposal, validation, simulation and memory-update calls reproducible. The backend language model used for the main EvoAgent runs was GPT-5.2 accessed through an OpenAI-compatible OpenRouter interface. Proposal and follow-up calls used `max_tokens=4096`; temperature, top-p and seed used the provider defaults. The run logs retained the model identifier, request metadata and returned text/tool calls. Outcome feedback entered the in-context lineage memory and reflection. Supplementary Table S16 summarizes the agent interface, including run prompts, tool schemas, parser rules and model-call metadata.

Table S16: LLM-agent implementation fields archived for reproducibility.

Component	Archived content
Model endpoint	GPT-5.2 accessed through an OpenAI-compatible OpenRouter endpoint; request date range, run identifier and model identifier retained with the run logs.
Embedding endpoint	Retrieval embeddings used <code>openai/text-embedding-3-small</code> through the OpenRouter OpenAI-compatible embedding endpoint; authentication was supplied through the run environment.
Decoding settings	Proposal and follow-up calls used <code>max_tokens=4096</code> , <code>tool_choice=auto</code> , streaming enabled and a 90s LLM timeout; temperature, top-p and seed used provider defaults. Tool and Maxwell timeouts were 180s and 900s, respectively. The retry sequence used a non-streaming call followed by a forced validation-tool call, with up to three constraint-repair attempts per round.
Run prompts	The reproducibility archive records the verbatim master input, rendered prompt context and returned model messages/tool calls after removal of credentials and machine-specific paths.
Prompt context	Current genotype, incumbent best score, recent lineage summary, active constraint feedback, retrieved physical priors and requested output fields.
Structured output	Natural-language diagnosis and rationale plus tool-call arguments containing proposed parameter values; the parser extracted numeric variables from the tool arguments and checked their completeness.
Tool schemas	Names, input fields and expected return fields for constraint validation, Maxwell evaluation, Adams evaluation, lineage-memory retrieval and logging tools.
Critic schemas	Magnetic-circuit, performance-trend, constraint-boundary and magnitude critic prompts, score range, confidence field and rejection criteria. The default critic backend was <code>openai/gpt-4o</code> , with temperature 0.3 and <code>max_tokens=500</code> .
Parsing and repair	Numeric extraction rules, bound clipping, invalid-output handling, repair prompt and discarded-proposal logging.
Lineage update	Instance-record schema, reflection trigger, promoted-rule format, transfer tag and prompt-context inclusion flag.

The prompt excerpts below give the task-level instructions used to start the design runs. The original run-level prompts were written in Chinese; English translations are reported here for readability. The reproducibility archive contains the verbatim Chinese master prompts and rendered per-round prompts, with credentials, machine-specific paths and transient provider metadata removed.

Table S17: Representative EvoAgent run-level prompts. The original prompts were written in Chinese and are reported here as English translations.

Prompt stage	Representative prompt text
Actuator run prompt	You are an electromagnetic-actuator optimization engineer and must autonomously complete multiple iterations. In each round, first restate the objective and feasible domain, propose candidate parameters, immediately call <code>validate_maxwell_design</code> , and use the returned errors and derived quantities to explain why the current settings were chosen and how the next adjustment should proceed. If <code>status=constraint_violation</code> , immediately revise the parameters and validate again. Only when validation returns <code>ok</code> should <code>run_maxwell_simulation</code> be called. Record logs, score, average magnetic flux density, <code>kb</code> , <code>pb</code> and related outputs, judge whether the best solution improves, and explicitly state why the round was adjusted in that way. Repeat until improvement plateaued or the evaluation budget ended. Whenever parameters are proposed, the order must be validation followed by simulation if feasible. Finally, output the global best design, a round-by-round summary and recommendations for further work.
Robot-leg run prompt	Optimize the robot-leg structure and follow a multi-round <code>validate_robot_design</code> and <code>run_robot_simulation</code> workflow.

During each run, the optimizer supplied the current design state, recent lineage records, retrieved design priors, active constraint margins and required structured proposal fields as context. Responses with complete parsed optimization variables and a feasible deterministic validation result entered the simulation workflow; the remaining proposals followed the configured repair or archival path.

Supplementary Note 7 Ablations of retrieval and lineage memory

The ablation experiments varied retrieval and lineage memory while holding the actuator evaluator and viability filters fixed. Each condition was run three times. The LLM-only and EvoAgent conditions reuse the three actuator repeats reported in the Fig. 2b comparison, and the retrieval-only and lineage-memory-only conditions report their own three repeated runs. Supplementary Table S18 lists the configuration definitions and the resulting summary.

These ablations separate proposal reasoning by the language model from externally stored design knowledge. The runs plotted in Supplementary Fig. S3 are the same three repeats summarized in Supplementary Table S18. For each run, the final cumulative score was recorded, and the median across the three runs was used as the primary summary statistic. The median scores were -11.52 for LLM-only, -15.72 for retrieval only, -16.93 for lineage memory only and -17.41 for EvoAgent.

Table S18: Retrieval and lineage-memory ablation summary for actuator optimization.

Configuration	Retrieval	Lineage	Median	Improvement
LLM-only	no	no	-11.52	–
Retrieval only	yes	no	-15.72	18.0%
Lineage-memory only	no	yes	-16.93	31.5%
EvoAgent	yes	yes	-17.41	37.0%

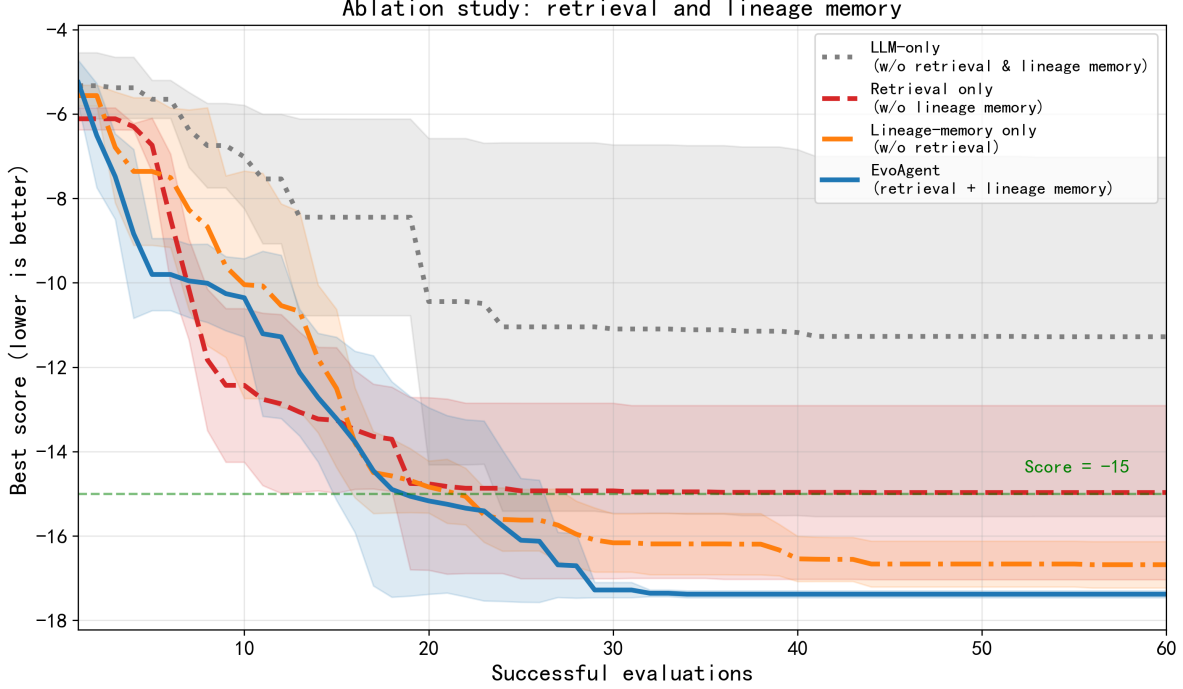


Figure S3: Retrieval and lineage-memory ablation for actuator optimization.

Supplementary Note 8 Transfer and expert-feedback protocols

Transfer experiments reused structured lineage memory across related actuator settings and topologies. Cross-parameter transfer changed the winding-wire diameter from 0.05 mm to 0.06 mm. This perturbation changes the attainable winding density within the same slot geometry and therefore tests whether rules learned from the tradeoff between slot depth, turn count, magnetic cross-section and saturation remain useful after a parameter-scale shift. With lineage transfer, the mean rounds to reach score thresholds of ≤ -7 , ≤ -8 and ≤ -9 were reduced relative to runs without transfer, corresponding to speedups of approximately 10.5-, 5.0- and 3.7-fold, respectively (Supplementary Fig. S4).

In the reported transfer tests, source-task lineage memory provided an end-to-end warm start. Transferred parameter priors informed the initial genotype in the target design regime, and retrieved physical rules guided subsequent mutations. The transfer results therefore measure the combined effect of memory-informed initialization and rule-guided mutation.

Cross-structure transfer reused EI-core experience in E-core optimization, where the magnetic topology changed while winding-space, saturation and package constraints remained relevant. Transferred lineage rules acted as reusable physical priors. In the E-core task, transfer reduced the mean rounds to reach score thresholds of ≤ -10 , ≤ -12 and ≤ -15 by approximately 8.0-, 10.8- and 9.4-fold, respectively (Supplementary Fig. S5).

In the EvoAgent (FB) condition, expert feedback entered as lineage-level guidance. The initial advice prioritized winding-turn density in early actuator exploration and specified that local yoke or bridge saturation should prompt consideration of larger arm/yoke thickness t_a or bridge thickness t_b .

EvoAgent generated candidate parameters from the current design state, lineage history, constraint margins and retrieved engineering priors. Expert feedback reduced the mean rounds to reach score thresholds of ≤ -10 , ≤ -12 and ≤ -15 by approximately 4.6-, 5.2- and 5.7-fold relative to EvoAgent without feedback (Supplementary Fig. S6).

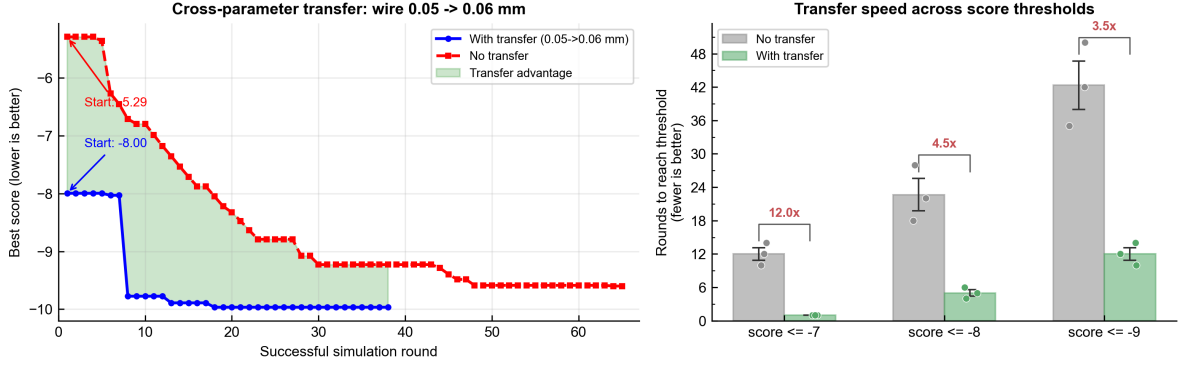


Figure S4: Cross-parameter transfer from 0.05 mm to 0.06 mm winding wire.

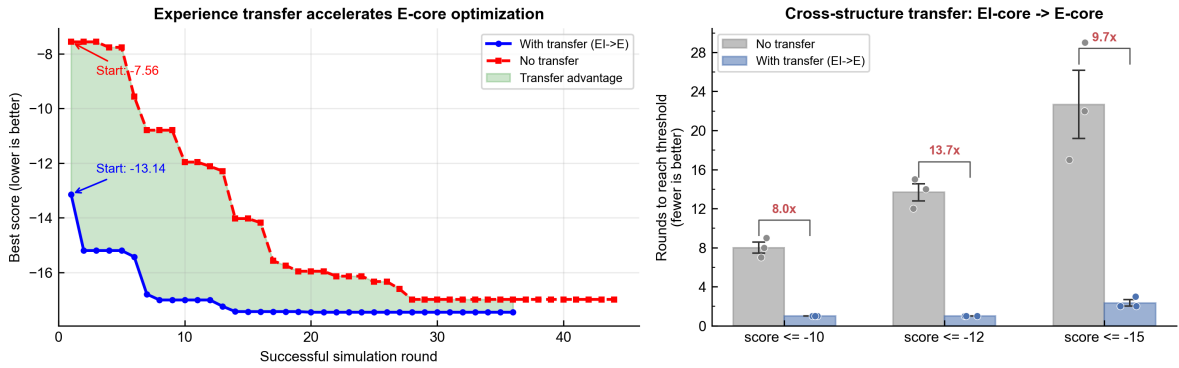


Figure S5: Cross-structure transfer from EI-core to E-core actuator optimization.

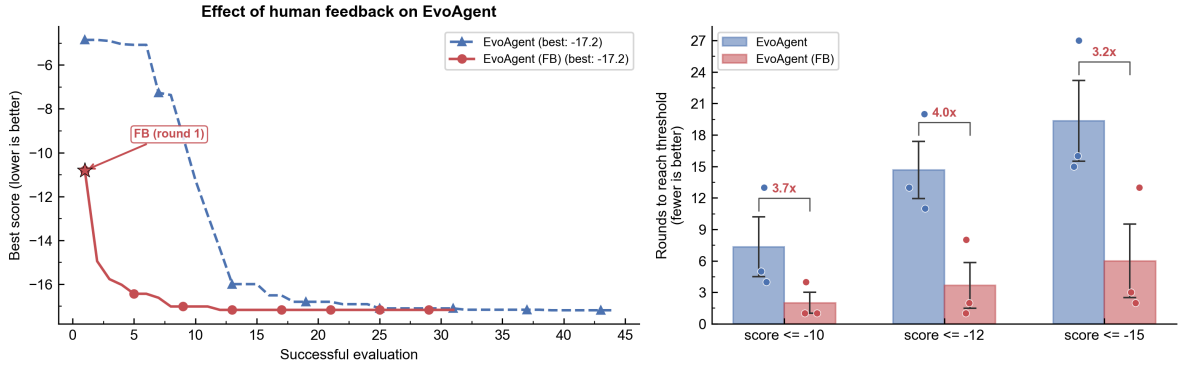


Figure S6: Effect of expert feedback on retrieval plus lineage memory and base configurations.

Supplementary Note 9 Hardware fabrication and robot assembly

Actuator assembly. The electromagnetic actuator was assembled from custom machined components. The coil was wound by machine to the target turn count. The E-core and I-plate were wire-cut from fixed-thickness silicon-steel sheets. The permanent magnets were custom cut, magnetized and made from NdFeB N52. During assembly, the permanent magnets were attached to the two sides of the E-core center beam by magnetic attraction, and the I-plate was also seated by magnetic attraction

to complete the designed magnetic path. The wound coil was then inserted through the E-core center beam. The assembly sequence is shown in Supplementary Fig. S7.

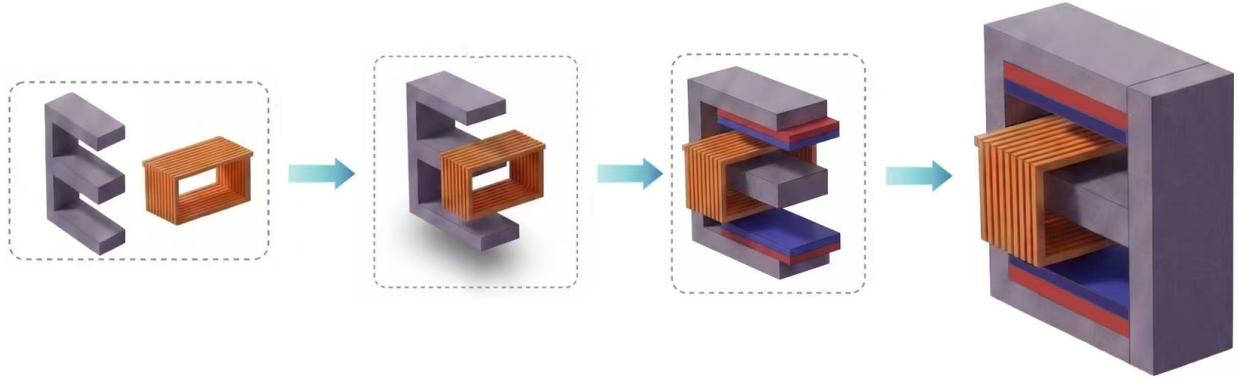


Figure S7: Assembly sequence of the electromagnetic linear actuator.

Robot structural assembly. After actuator assembly, the actuator module was integrated with the robot leg structure. The front-leg system comprised the front leg, actuator stator components, battery and control module. The E-core was joined to the front leg using a light interference fit of 0.05 mm. The battery was bonded above the front leg and connected to the controller and coil. The rear-leg system comprised the rear leg and actuator mover component, namely the coil. The rear-leg structure reserved a coil mounting position with a 0.1 mm interference fit, and the coil was fixed to the rear-leg component using ultraviolet-curable adhesive (ergo 8500). The robot assembly sequence is shown in Supplementary Fig. S8.

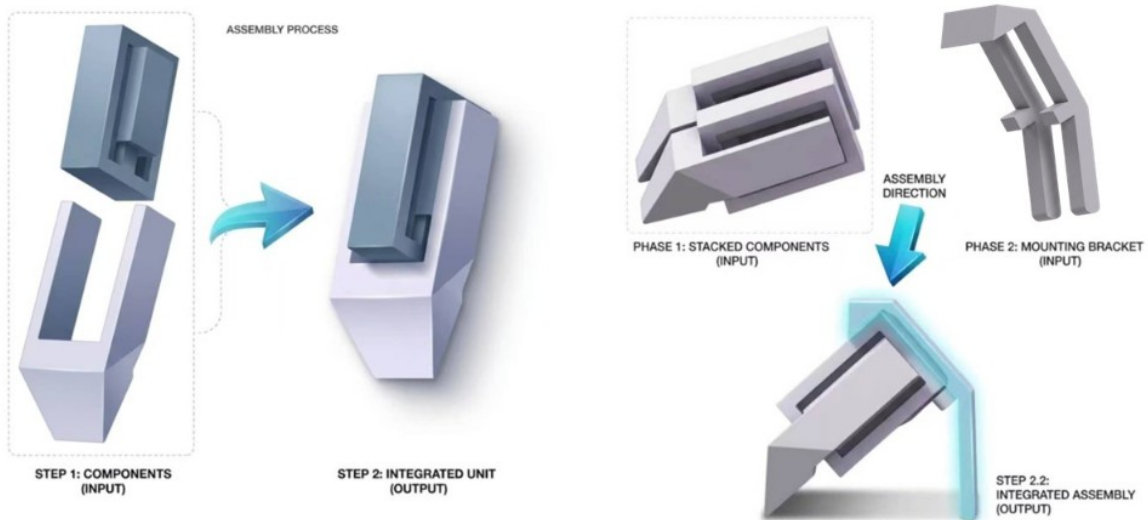


Figure S8: Robot structural assembly sequence. The figure shows the transition from individual body components to the integrated actuator–leg assembly used in the untethered robot.

Wireless control electronics. The wireless control circuit used a Bluetooth-controlled inverter architecture. The microcontroller module was an HJ-840 module from Tangshan Hongjia Electronic Technology, with dimensions of 6.2 mm×7 mm×0.9 mm and mass of approximately 300 mg. The module includes a Bluetooth antenna, operates from 1.7 V to 5.5 V, and provides 48 GPIO interfaces. Motor driving used an HT8835ARDZ dual H-bridge driver from Shenzhen Haitianxin Microelectronics. The H-bridge package is 2 mm×3 mm, has a mass of approximately 150 mg, supports PWM control and can supply up to 1.5 A per bridge channel. An RS3236-3.3YUTDN4 linear regulator provided the 3.3 V supply rail. The regulator package is 0.95 mm×0.95 mm×0.34 mm and has a mass of approximately 9 mg. To reduce controller footprint while leaving clearance for the Bluetooth antenna, the PCB used a two-layer layout with the microcontroller on one side and the H-bridge and regulator circuits on the other side.

Fabricated robot dimensions. The geometric definitions used by the validator are shown in Supplementary Fig. S1. The fabricated dimensions used for the final prototype are summarized below.

Table S19: Representative geometric parameters of the fabricated EvoAgent robot.

Param.	Value	Param.	Value	Param.	Value	Param.	Value
l_r	11.24 mm	h_1	4.7 mm	l_1	7.56 mm	l_2	6 mm
h_r	9.96 mm	l_3	2.09 mm	l_4	5.38 mm	l_5	1.87 mm
w_r	7.4 mm	w_1	3.3 mm	w_2	4.5 mm	w_3	3.4 mm
α	96.83°	w_4	2 mm	d_1	0.6 mm	d_2	2.2 mm
β	76.63°	d_3	0.8 mm	d_4	0.75 mm	d_5	1 mm
θ	37°	h_2	5.7 mm				

Supplementary Note 10 Hardware measurement statistics

The actuator thrust comparison used three independently fabricated EvoAgent actuators and three independently fabricated Darwinian-baseline actuators. The statistical unit for the actuator measurements was the device-level fitted thrust coefficient.

Static thrust was measured on the setup shown in Supplementary Fig. S9. The platform comprised an analytical balance, a custom holding fixture and a programmable DC power supply. The coil was fixed by the holding fixture and remained stationary during current loading, while the iron-core assembly acted as the force-output component and contacted the measurement platform of the balance. Before each measurement, the balance was zeroed at zero current to remove the static contribution from the fixture, prototype weight and initial contact state. The relative position between the actuator and balance platform was then adjusted until a stable vertical contact state was obtained. Direct currents from 10 mA to 50 mA were applied to the coil, and the balance-readout change was used to quantify the static thrust transmitted by the iron-core assembly. The same coil clamping, contact condition, baseline subtraction, current-loading protocol and thrust conversion were used for all actuator prototypes. A device-level thrust coefficient was then fitted for each actuator, and group-level means and s.e.m. were computed across fabricated actuators, with fabricated actuator as the statistical unit.

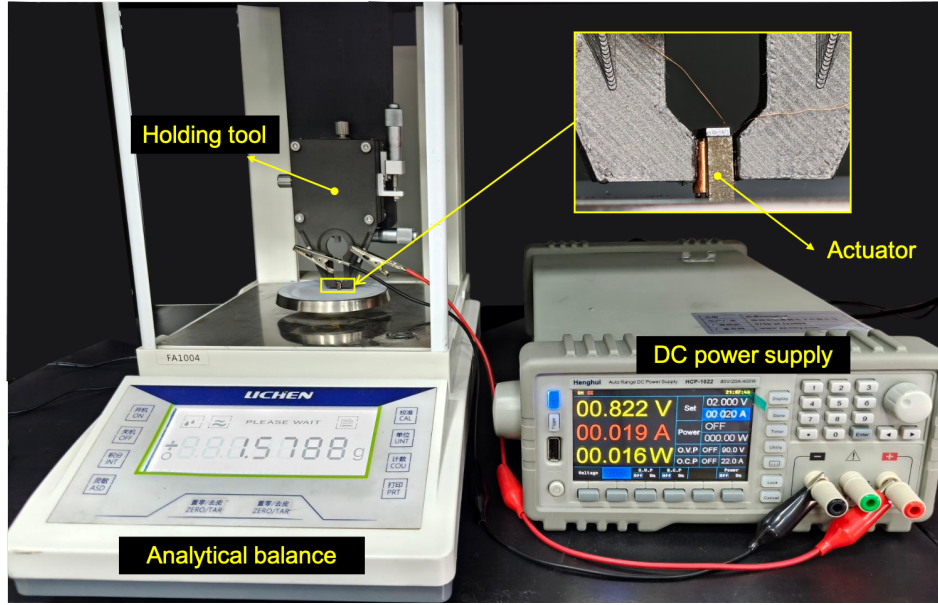


Figure S9: Static-thrust measurement setup for the electromagnetic linear actuator. The setup includes an analytical balance, a custom holding fixture and a programmable DC power supply. The coil is fixed by the holding fixture, and the iron-core assembly transmits the electromagnetic force to the balance platform during DC current loading.

Robot locomotion was measured on the setup shown in Supplementary Fig. S10. The setup comprised an LED panel light, an adjustable camera support, a smartphone camera, a control computer, a calibration ruler and a PVC test substrate. The LED light provided uniform illumination, and the camera recorded the robot motion in the test plane. The calibration ruler was placed parallel to the main travel direction to convert image pixels to physical length. Before testing, the robot was placed on the PVC substrate with a prescribed initial pose, and the camera view was adjusted to include the robot trajectory, ruler and test plane. The control computer set the wireless driving parameters, including frequency, duty cycle and input current. Each trial used a 1 s motion window. Net forward displacement was extracted from the video or image sequence after spatial calibration and normalized by the 10 mm body length to report forward speed in body lengths per second (BL s^{-1}); conditions without identifiable forward motion were assigned zero. The robot operating map used a 6×5 frequency–current grid, and each condition was measured three times for both EvoAgent and Darwinian-baseline robots. The map mean was obtained by first averaging the three repeats within each condition and then averaging across the 30 condition means.

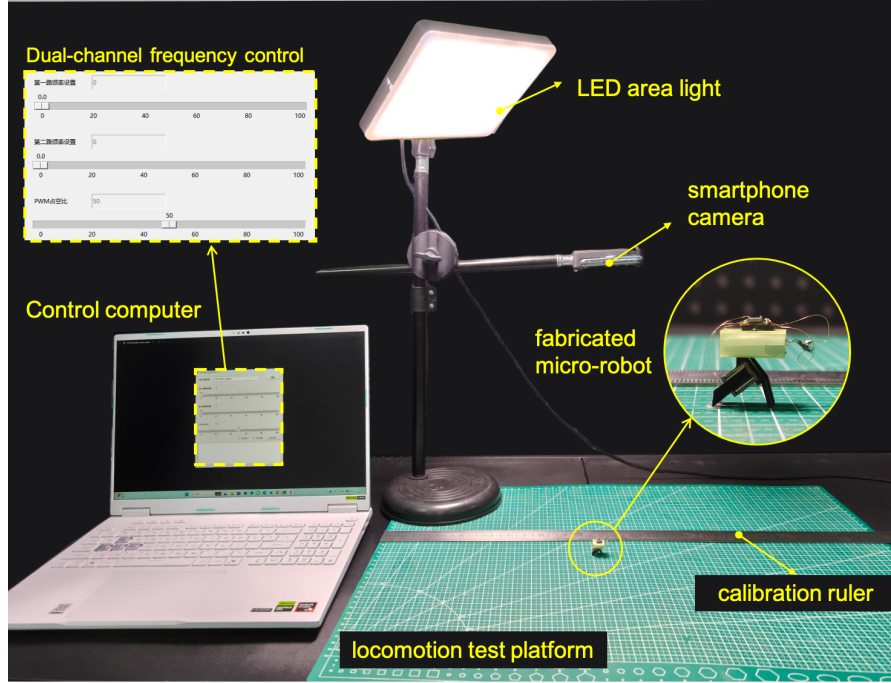


Figure S10: Motion-performance measurement setup for the untethered millimeter-scale robot. The setup includes an LED panel light, a smartphone camera mounted on an adjustable support, a control computer, a calibration ruler and a PVC test substrate.

Table S20: Hardware summary statistics used for Fig. 4.

Comparison	Statistical unit	EvoAgent	Darwinian baseline
Actuator thrust coefficient	fabricated actuator	0.663 mN mA^{-1} , $n = 3$	0.284 mN mA^{-1} , $n = 3$
Maximum forward speed	single 1 s trial	6.5 BL s^{-1}	—
Robot operating-map speed	condition mean	2.76 BL s^{-1} over 30 condition means	0.43 BL s^{-1} over 30 condition means
Operating-map sampling and summary	frequency–current grid	6×5 grid (30 conditions); 3 repeats per condition; condition means averaged for the map summary	

Supplementary Note 11 Literature comparison of untethered robots

The table below records the literature comparator set used to support the state-of-the-art robot comparison in Fig. 4f of the main text. Inclusion requires untethered operation with onboard electrical or chemical energy; platforms requiring a power or control tether are excluded. Metrics are reported as stated in the corresponding source papers or figure source data. The speed column reports body-length-normalized locomotion speed where available.

Table S21: Onboard-powered, untethered literature comparator set for Fig. 4f. Platforms requiring power or control tethers are excluded.

Ref.	Robot	Year	Size	Mass	Speed	Turning	Payload
[S1]	BHMbot	2024	19–20 mm	1.76 g	17.5 BL/s	750 deg/s [†]	>2 g
[S2]	BHMbot-B	2024	15–20 mm	1.89 g	18.0 BL/s	120 deg/s	32× BW
[S3]	Soft microrobot	2025	20 mm	1.12 g	4.8 BL/s	280 deg/s	53,000× BW
[S4]	Insect-scale soft robot	2023	24 mm	380 mg	2.5 BL/s	18 deg/s	1.81 g
[S5]	S2worm	2022	41 mm	4.34 g	6.7 BL/s	not reported	none
[S6]	S2worm-G	2024	40 mm	4.71 g	18.8 BL/s	133 deg/s	none
[S7]	HAMR-F	2018	45 mm	2.8 g	3.8 BL/s	not reported	none
[S8]	RoACH	2008	30 mm	2.4 g	1.0 BL/s	controlled; rate not reported	none
[S9]	RoBeetle	2020	15 mm	88 mg	≈0.05 BL/s	none	2.61× BW
[S10]	5 mm crawler	2022	5 mm	46.4 mg	1.18 BL/s	none	none
[S11]	DEAnsect	2019	40 mm	≈1 g	0.75 BL/s	none	none
[S12]	Multisensor terrestrial robot	2025	42 mm	not reported	1.72 BL/s	not reported	20 g
This work	EvoAgent robot	2026	10 mm	1 g	6.5 BL/s	356 deg/s	>5× BW

[†]Derived from the maximum reported 300° turn completed in 0.4 s; it is used only for the angular-speed colour encoding in Fig. 4f. The source paper separately reports turning centripetal acceleration.

Supplementary Movies

Four movies accompany the main text. All movies show the fabricated 10 mm-long, 1 g untethered robot driven by the onboard wireless controller on the PVC test substrate described above. Each legend states the pulse-width-modulation (PWM) duty cycle and the drive frequency of the left and right actuator channels.

Supplementary Movie 1. High-speed imaging of robot locomotion at three drive frequencies. Both actuator channels are driven at a 50 % duty cycle, and the robot is recorded successively at 40 Hz, 20 Hz and 10 Hz. The sequences resolve front- and rear-leg motion and the resulting body displacement, and show how the leg kinematics change with drive frequency.

Supplementary Movie 2. Forward locomotion of the untethered robot. Both actuator channels are driven at 20 Hz with a 40 % duty cycle, and the robot advances along the calibrated travel direction.

Supplementary Movie 3. Turning of the untethered robot, recorded from a top view. The left actuator channel is driven at 20 Hz with a 40 % duty cycle while the right channel is switched off, so that the energized side sweeps the robot about the unpowered side.

Supplementary Movie 4. Payload transport by the untethered robot. Both actuator channels are driven at 20 Hz with a 40 % duty cycle, additional mass is attached to the robot body, and forward locomotion remains stable up to a payload equal to 5.2 times the robot weight.

Supplementary References

- [S1] Zhiwei Liu, Wencheng Zhan, Xinyi Liu, Yangsheng Zhu, Mingjing Qi, Jiaming Leng, et al.. A wireless controlled robotic insect with ultrafast untethered running speeds. *Nature Communications* **15**, 3815 (2024). <https://doi.org/10.1038/s41467-024-47812-5>.
- [S2] Xian Yu, Wencheng Zhan, Zhiwei Liu, Lizhao Wei, Wei Shen, Ruide Yun, et al.. Forward and backward control of an ultrafast millimeter-scale microrobot via vibration mode transition. *Science Advances* **10**, eadr1607 (2024). <https://doi.org/10.1126/sciadv.adr1607>.
- [S3] Yichuan Wu, Lai Cao, Guobin Lu, Peng Wang, Longqi Ran, Bei Peng. Untethered soft microrobot driven by a single actuator for agile navigations. *Nature Communications* **16**, 6842 (2025). <https://doi.org/10.1038/s41467-025-61810-1>.
- [S4] Zicong Miao, Jiaming Liang, Huimin Chen, Jiangfeng Lu, Xiang Sun, Ying Liu, et al.. Power autonomy and agility control of an untethered insect-scale soft robot. *Soft Robotics* **10**, 749–759 (2023). <https://doi.org/10.1089/soro.2021.0201>.
- [S5] Yide Liu, Yanhong Chen, Bo Feng, Dongqi Wang, Taishan Liu, Haofei Zhou, et al.. S²worm: A fast-moving untethered insect-scale robot with 2-DoF transmission mechanism. *IEEE Robotics and Automation Letters* **7**, 6758–6765 (2022). <https://doi.org/10.1109/LRA.2022.3176435>.
- [S6] Yide Liu, Bo Feng, Tianlun Cheng, Yanhong Chen, Xiyan Liu, Jiahang Zhang, et al.. Singularity analysis and solutions for the origami transmission mechanism of fast-moving untethered insect-scale robot. *IEEE Transactions on Robotics* **40**, 777–796 (2024). <https://doi.org/10.1109/TR0.2023.3338949>.

- [S7] Benjamin Goldberg, Raphael Zufferey, Neel Doshi, Elizabeth F. Helbling, Griffin Whittredge, Mirko Kovac and Robert J. Wood. Power and control autonomy for high-speed locomotion with an insect-scale legged robot. *IEEE Robotics and Automation Letters* **3**, 987–993 (2018). <https://doi.org/10.1109/LRA.2018.2793355>.
- [S8] A. M. Hoover, E. Steltz and R. S. Fearing. RoACH: An autonomous 2.4-g crawling hexapod robot. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 26–33 (2008). <https://doi.org/10.1109/IRoS.2008.4651149>.
- [S9] Xiufeng Yang, Longlong Chang and Nestor O. Perez-Arancibia. An 88-milligram insect-scale autonomous crawling robot driven by a catalytic artificial muscle. *Science Robotics* **5**, eaba0015 (2020). <https://doi.org/10.1126/scirobotics.aba0015>.
- [S10] Yangsheng Zhu, Mingjing Qi, Zhiwei Liu, Jianmei Huang, Dawei Huang, Xiaojun Yan, et al.. A 5-mm untethered crawling robot via self-excited electrostatic vibration. *IEEE Transactions on Robotics* **38**, 719–730 (2022). <https://doi.org/10.1109/TR0.2021.3088053>.
- [S11] Xiaobin Ji, Xinchang Liu, Vito Cacucciolo, Matthias Imboden, Yoan Civet, et al.. An autonomous untethered fast soft robotic insect driven by low-voltage dielectric elastomer actuators. *Science Robotics* **4**, eaaz6451 (2019). <https://doi.org/10.1126/scirobotics.aaz6451>.
- [S12] Baekgyeom Kim, Haram Lee, Seongjun Lee, Seokhaeng Huh, Jaekwan Ryu and Je-Sung Koh. Insect-scale terrestrial robot capable of untethered environmental monitoring with multisensors. *IEEE/ASME Transactions on Mechatronics* **30**, 7395–7405 (2025). <https://doi.org/10.1109/TMECH.2025.3583215>.