

LASER: TOWARDS DIVERSIFIED AND GENERALIZABLE ROBOT DESIGN WITH LARGE LANGUAGE MODELS

Anonymous authors

Paper under double-blind review

ABSTRACT

Recent advances in Large Language Models (LLMs) have stimulated a significant paradigm shift in evolutionary optimization, where hand-crafted search heuristics are gradually replaced with LLMs serving as intelligent search operators. However, these studies still bear some notable limitations, including a challenge to balance exploitation with exploration, often leading to inferior solution diversity, as well as poor generalizability of problem solving across different task settings. These unsolved issues render the prowess of LLMs in robot design automation largely untapped. In this work, we present LASeR – Large Language Model-Aided Evolutionary Search for Robot Design Automation. Leveraging a novel reflection mechanism termed DiRect, we elicit more knowledgeable exploratory behaviors from LLMs based on past search trajectories, reshaping the exploration-exploitation tradeoff with dual improvements in optimization efficiency and solution diversity. Additionally, with evolution fully grounded in task-related background information, we unprecedentedly uncover the inter-task reasoning capabilities of LLMs, facilitating generalizable design processes that effectively inspire zero-shot robot proposals for new applications. Our simulated experiments on voxel-based soft robots showcase distinct advantages of LASeR over competitive baselines.

1 INTRODUCTION

In recent years, Large Language Models (LLMs) have demonstrated remarkable reasoning, decision making, and generalization capabilities (Achiam et al., 2023; Touvron et al., 2023; Team et al., 2023; Team, 2023), sparking a flurry of research interest in their application to optimization problems. Earlier efforts embarked on leveraging LLMs to aid traditional search heuristics within evolutionary algorithms (EAs), such as selecting parent solutions for mutation and crossover (Liu et al., 2024a; Ye et al., 2024) or serving as surrogate model and candidate sampler in Bayesian Optimization (Liu et al., 2024b). More and more recent studies have explored the use of LLMs as “intelligent search operators”. By receiving previously found solutions through prompts, LLMs effectively draw upon their in-context learning and pattern completing abilities to iteratively propose improved candidate solutions (Brahmachary et al., 2024; Huang et al., 2024b; Yang et al., 2024; Morris et al., 2024; Romera-Paredes et al., 2024; Lange et al., 2024). Such LLM-aided evolutionary frameworks have shown great promise in minimizing reliance on handcrafted search heuristics, facilitating convenient problem specification in natural language and rendering evolutionary processes more interpretable. To date, they have showcased proficiency in classic optimization problems such as the Traveling Salesman Problem and numerical functions, (Liu et al., 2024a; Brahmachary et al., 2024; Huang et al., 2024a), as well as real-world scenarios spanning code generation (Morris et al., 2024; Romera-Paredes et al., 2024), robotic control (Lange et al., 2024), protein design (Tran & Hy, 2024), etc.

Despite the promising results, we contend that existing studies exhibit two major limitations. For one, as many of them have noted that LLMs often struggle to balance exploration and exploitation and yield inferior solution diversity (Huang et al., 2024b; Tran & Hy, 2024), only expedient measures have been taken to address this issue, including adjustments to the temperature parameter (Yang et al., 2024; Liu et al., 2024a; Pluhacek et al., 2024; Ma et al., 2024) or utilizing pre-existing natural selection techniques such as binary tournament selection and “island models” (Qiu et al.,

2024; Romera-Paredes et al., 2024). It remains to be investigated whether the reasoning capabilities of LLMs could be further harnessed to guide more intelligent exploratory behaviors in the search space. For the other, current LLM-aided evolutionary approaches generally lack a strong connection to the specific nature of real-world problems, which leads to suboptimal performances and solutions that can not generalize well.

Recently, LLMs have also made their way into the realm of robot design automation, which represents a persistent challenge in modern robotics that aims to evolve robot morphology with minimal human intervention (Hu et al., 2022; 2023; Song et al., 2024a). However, related work is sparse and only represent rudimentary attempts. To our best knowledge, the only pertinent studies are Zhang (2024), Qiu et al. (2024) and Lehman et al. (2023). While Zhang (2024) utilizes LLMs to tune the hyperparameters of traditional EAs, the latter two pioneer the use of LLMs as search operators for robot design. Nonetheless, they bear the same limitations as listed above, which greatly hinder the application of LLMs to robot design automation. In particular, with growing interest in soft robots due to their versatility and biomimetic properties, their vast design spaces and intricacy of interaction dynamics among body parts cause existing search algorithms to generally fall short, highlighting the need for more judicious exploration that navigate a variety of design options while ensuring progressive enhancement in functionality. (Bhatia et al., 2021; Shah et al., 2021; Song et al., 2024a; Saito & Oka, 2024). Furthermore, as it is common to have access to a repository of pre-designed robots from related tasks when designing for new applications, it is highly relevant to explore the inter-task reasoning capabilities of LLMs to facilitate positive transfer of prior design experience, thus fostering more generalizable design processes.

To address the aforementioned limitations, here we propose **LASeR** – **L**arge Language Model-**A**ided Evolutionary **S**earch for **R**obot Design Automation. LASeR distinguishes itself from previous LLM-aided evolutionary frameworks with a more delicate exploration strategy and generalizable optimization processes. Specifically, we present a novel *Diversity Reflection Mechanism* termed **Di-Rect**, which strategically instruct an LLM to reflect upon previously generated designs and suggest viable modifications to enhance diversity while preserving essential functional substructures. This mechanism thus fosters more knowledgeable exploratory behaviors that closely align with task objectives. Furthermore, by exploiting the abundant descriptive information available in robotic tasks, we not only yield substantially accelerated convergence to high-performing designs, but also unprecedentedly uncover the potential of LLMs to reason across different tasks and assimilate prior design experience for zero-shot robot proposals in new tasks.

To summarize, our contributions are as follows: (i) By interleaving evolutionary processes with diversity-oriented reflective thinking, we reshape the exploration-exploitation tradeoff of LLM-aided evolution with simultaneous improvements in solution diversity and optimization efficiency. The former is particularly relevant for enhancing the robustness of robotic systems in volatile environments. (ii) With evolution firmly grounded in the background information of optimization tasks, we unlock the inter-task reasoning capabilities of LLMs in evolutionary computation, hopefully inspiring future work on further promoting generalizability of LLM-aided evolutionary processes across different problem settings. (iii) By fully unleashing the prowess of LLMs for robot design automation, we also aim to inspire future work that synergizes both design and control with LLMs, achieving closed-loop development of embodied agents.

2 RELATED WORK

Large Language Models as Evolutionary Search Operators. Large Language Models (LLMs) represent a class of deep generative neural networks comprising billions or trillions of parameters and pretrained on web-scale textual data. In recent years, LLMs have demonstrated impressive reasoning, decision making, and generalization capabilities (Achiam et al., 2023; Touvron et al., 2023; Team et al., 2023; Team, 2023), which have sparked a flurry of research into exploiting them for optimization problems (Huang et al., 2024b; Wu et al., 2024). By receiving history search trajectories from the prompt (or context), LLMs have demonstrated effectiveness as pattern completion engines (Mirchandani et al., 2023), proposing improved solutions and facilitating evolutionary optimization through iterative interactions. Moreover, LLMs are adept at conditioning problem-solving processes on various kinds of prior knowledge expressed in natural language, without needing tedious mathematical formulations (Song et al., 2024b). All these favorable attributes position LLMs as promising

108 substitutes for the manually designed search heuristics in traditional evolutionary algorithms (EAs),
109 acting as novel, *intelligent* search operators. Since Lehman et al. (2023) introduced this LLM-
110 aided evolutionary paradigm, subsequent studies have extended its methodology and showcased its
111 proficiency in classic optimization tasks like the Traveling Salesman Problem (TSP) and numerical
112 functions (Liu et al., 2024a; Brahmachary et al., 2024; Huang et al., 2024a), as well as practical
113 problems spanning prompt optimization (Guo et al., 2023; Yang et al., 2024), code generation (Mey-
114 erson et al., 2023; Morris et al., 2024; Romera-Paredes et al., 2024), robotic control (Lange et al.,
115 2024), etc. However, we argue that the use of LLMs as search operators is still in its early stage,
116 with much of their potential untapped. Notably, existing studies have focused solely on single-task
117 optimization, overlooking the intriguing possibility of LLMs to transfer experience across different
118 tasks. Additionally, although LLMs have been shown to trail behind traditional EAs in balancing ex-
119 ploration and exploitation (Huang et al., 2024a; Tran & Hy, 2024), this nuanced aspect has received
120 limited attention from previous research. We aim to tackle these limitations in this work.

121 **Robot Design Automation.** As Artificial Intelligence (AI) continues to revolutionize academia
122 and industry, there is an increasing focus on integrating the perceptual and planning capabilities
123 of multi-modal foundation models into various physical embodiments capable of interacting with
124 their environments – a research field known as *Embodied AI* (Roy et al., 2021; Liu et al., 2024c).
125 These advancements highlight the significance of autonomous robot design. Earlier works on robot
126 design automation relied on traditional evolutionary algorithms and primarily targeted rigid robots
127 (Sims, 1994; Chocron & Bidaud, 1997; Leger, 2012; Wang et al., 2019). In recent years, modular
128 soft robots have garnered broad attention due to their flexibility, expressiveness, and biomimetic
129 characteristics (Hiller & Lipson, 2011; Bhatia et al., 2021; Medvet et al., 2021). However, these
130 advantages are accompanied by a combinatorially vast design space, necessitating more efficient
131 search algorithms (Cheney et al., 2014). Consequently, an emerging line of research resorts to the
132 estimation-of-distribution algorithms (EDAs) to enhance sample efficiency by explicitly tracking the
133 distribution of high-performing robot designs. These approaches further leverage deep generative
134 models, such as Generative Adversarial Networks (GANs; Goodfellow et al., 2020) and Variational
135 Autoencoders (VAEs; Kingma, 2013), to bolster the representational capacity of EDAs (Hu et al.,
136 2022; Song et al., 2024a). Despite their promising results, these models still require problem-specific
137 mathematical formulation and neural architecture design, which is highly dependent on domain
138 expertise and poorly generalizable. In this respect, Large Language Models, with their strong in-
139 context learning abilities and extensive prior knowledge, [hold the promise to transform the robotic
140 design process \(Stella et al., 2023\)](#). Nevertheless, the exploration of LLMs in this respect is sparse
141 and warrants further investigation (Lehman et al., 2023; Zhang, 2024; Qiu et al., 2024). [While our
142 work is based on simulation, we note that there is ongoing research on the realization of soft robotics
143 in the physical world, using polymers with pneumatic chambers \(Kriegman et al., 2020b; Legrand
144 et al., 2023\) or even self-replicating cells \(Kriegman et al., 2020a; 2021\) and continually narrowing
145 the sim-to-real gap. We believe that with the collective efforts of material scientists, computer
146 scientists, \(bio\)mechanical engineers, etc., soft robotics would see rapid advances and finds its way
147 to everyday life in the near future.](#)

147 3 LASER: LLM-AIDED EVOLUTIONARY SEARCH FOR ROBOT DESIGN 148 AUTOMATION 149

150 In this section, we first present an overview of our algorithm, and then delve into the details of our
151 prompt design and the novel *Diversity Reflection Mechanism*. Subsequently, we describe how LLMs
152 can be instructed to facilitate effective knowledge transfer across different tasks, followed by a brief
153 introduction to our fitness evaluation protocols.
154

155 3.1 ALGORITHM FRAMEWORK 156

157 As illustrated in Figure 1(a) and detailed by Algorithm 1 in Appendix R, we integrate an LLM into
158 the bi-level optimization framework commonly employed in robot design automation. Specifically,
159 the inner loop optimizes a controller for each robot morphology through reinforcement learning,
160 with the resulting task performance serving as the fitness evaluation. The outer loop evolves a
161 population of robot morphologies by carrying out natural selection and generating new offspring
solutions in each generation. Here, instead of traditional evolutionary algorithms (EAs) that rely on

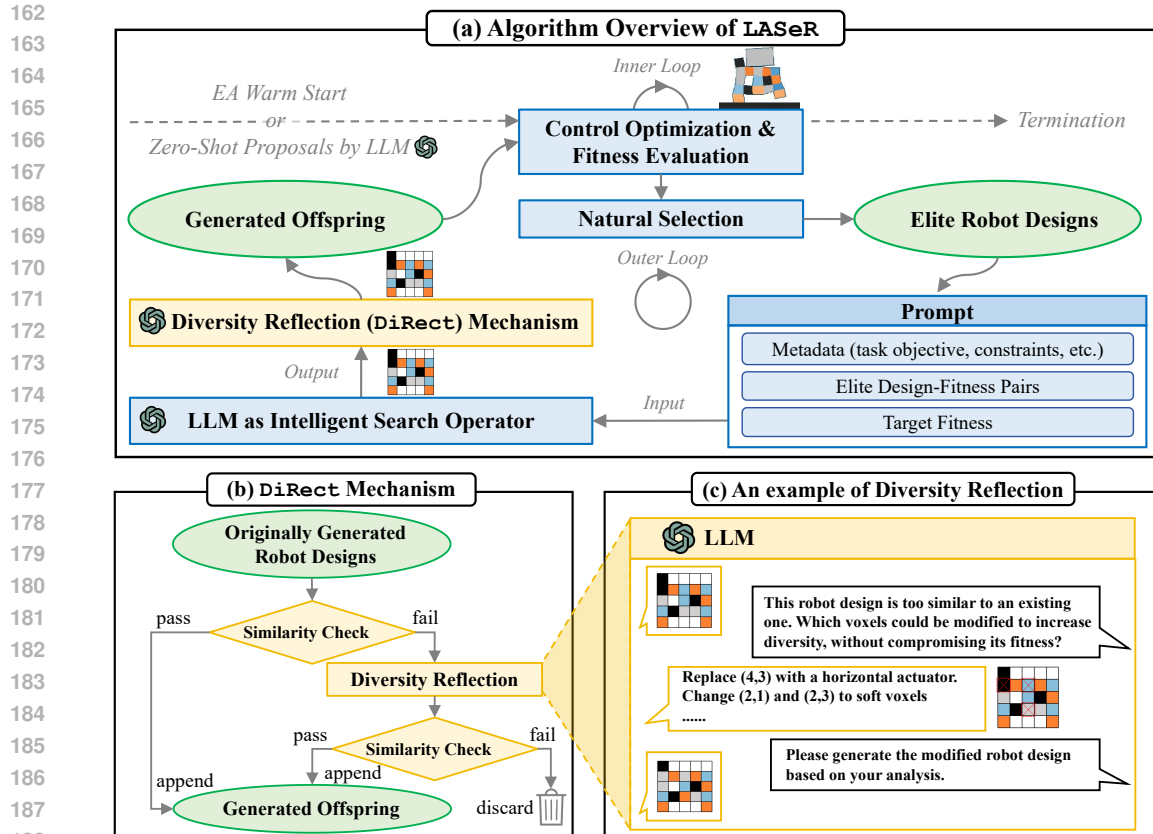


Figure 1: (a) algorithm overview of LAsER; (b) the Diversity Reflection (DiRect) Mechanism; (c) an example illustrating how Diversity Reflection works on a carrying robot. The illustration takes voxel-based soft robots (VSRs; Bhatia et al., 2021) as an example. Each VSR is represented as a two-dimensional material matrix, where each entry, ranging from 0 to 4, represents the material type at the corresponding position. \blacksquare , \square , \blacksquare , \blacksquare and \square denote rigid voxels, soft voxels, horizontal actuators, vertical actuators and empty voxels, respectively.

manually designed search heuristics for generating offspring, an LLM is properly prompted to be our search operator. This is achieved by providing the LLM with previously evaluated robots and various kinds of metadata as context. However, we still bootstrap the evolutionary process with a few generations of conventional EAs before letting the LLM play its part, as this warm start would provide the LLM with an initial momentum (i.e., improving directions) to build upon. Evolution terminates when a maximum number of robot evaluations is reached. We term our approach LAsER, short for LLM-Aided Evolutionary Search for Robot Design Automation. The following subsections elaborate on the key components of LAsER. Note that while the introduction takes voxel-based soft robots (VSRs) proposed in Bhatia et al. (2021) as an example, our approach is readily applicable to other robot types. For a detailed introduction to VSRs, please refer to Bhatia et al. (2021).

3.2 PROMPT DESIGN

As discussed in Section 3.1, the LLM is prompted to propose new offspring designs in each generation. As effective prompt design is crucial to elicit desired responses from LLMs, we craft a well-structured prompt comprising the following essential components:

- **Task-related metadata:** this part includes various auxiliary information to in-context adapt the LLM to serve as a search operator, including task objectives, descriptions of simulation environment, constraints on robot designs, etc. **This component is largely derived from the official documents of EvoGym (Bhatia et al., 2021), with minimal modifications.** This

metadata is largely missing in previous studies. We would showcase in our experiments that by fully grounding the evolutionary process on task-related metadata, we not only achieve more efficient design optimization, but also elicit inter-task reasoning capabilities from LLMs that greatly boost the generalization of design experience.

- Elite design-fitness pairs: this part includes robot designs that have survived natural selection, each accompanied by its fitness score. These designs are sorted in ascending order of fitness, so as to make it easier for the LLM to infer continuation patterns and construct potentially improved designs.
- Target fitness: we instruct the LLM to extrapolate from the sequence of elite designs and generate new designs that meet a pre-specified fitness score (e.g. 120% of current maximal fitness). This is referred to as the “just-ask” query in previous studies (Lim et al., 2024).

Additionally, we include some basic requirements into the system prompt to better align the behavior of the LLM with our intention. For instance, the LLM is strictly demanded to output new robot designs in numpy array format, enclosed between `<solution>` and `</solution>`, to allow for easier parsing. Even so, LLMs are still not guaranteed to generate valid robot designs in every interaction. To keep runtime and cost under control, we set an upper limit on interactions; once the limit is reached, we revert to a conventional EA to generate the remaining offspring. For full prompts please refer to Appendix A.

3.3 DiRECT: DIVERSITY REFLECTION MECHANISM

The exploration-exploitation tradeoff is a longstanding dilemma in evolutionary computation. Properly timed exploratory behavior would contribute to search processes that not only are less prone to getting stuck in local optima, but also produce more diversified solutions. The latter property is particularly relevant for developing robotic systems in dynamic environments, where first-choice robots, once fail, must be immediately replaced with alternatives. However, it has been shown that LLMs often struggle to balance exploration and exploitation, yielding inferior solution diversity to existing EAs (Huang et al., 2024a; Tran & Hy, 2024). While many previous studies addressed this issue by tuning the temperature parameter of LLMs or resorting to diversity-preserving selection techniques, in this work we introduce a novel *Diversity Reflection Mechanism* (DiRect) that leverages the reasoning capabilities of LLMs to guide exploratory behaviors.

The idea of DiRect is straightforward. As depicted in Figure 1(b), for each newly proposed robot design, we assess its similarity to previously evaluated designs with a probability p . The new design is said to fail the similarity check if it shares more than s voxels with at least one existing robot. In this case, the DiRect mechanism is triggered. Specifically, the LLM is first prompted to suggest modifications to voxels that could enhance variability without compromising fitness, with existing high-performing examples as reference. The LLM is then asked to return the modified robot design according to these suggestions. In section 4 we show that this reflection mechanism fosters more beneficial exploratory behavior in the search space, leading to more diversified robot designs while maintaining relevance to the task objective. Figure 1(c) displays a specific example where DiRect helps to modify a newly proposed carrying robot. [We include general principles for choosing the similarity threshold \$s\$, supported by experimental evaluations, in Appendix N.](#)

3.4 LLM FOR INTER-TASK KNOWLEDGE TRANSFER

The algorithm described thus far treats the robot design of each task independently, starting with a randomly initialized population. However, it is often the case that we already have access to a repository of pre-designed robots from existing tasks when designing for a new one. Under this circumstance, leveraging the prior design experience in one way or another would hopefully afford a boost in sample efficiency. However, to discern functional substructures from highly abstract robot morphologies (e.g. voxel-based soft robots), as well as to speculate which substructures will benefit a new task, poses a major challenge to humans. Our work represents a pioneering effort to exploit the reasoning capabilities of LLMs for this purpose. Specifically, by specifying the characteristics of task A and B, along with a collection of high-performing robot designs from task A, we instruct an LLM to analyze the similarities and differences between the two tasks and infer potentially favorable substructures (such as specific patterns of voxel assembly) for task B. Based on this analysis, the

LLM then proposes robot designs for task B, thus enabling *zero-shot* robot proposals (where “zero-shot” means that no evaluated robot samples from the new task are required). These robot proposals can then serve as an *informative* initialization to initiate further design search for task B.

3.5 FITNESS EVALUATION

In this work, we use the Proximal Policy Optimization (PPO) algorithm to optimize a separate controller for each robot design. The PPO algorithm enhances conventional gradient-based reinforcement learning algorithms by incorporating importance sampling into gradient estimation, allowing for the reuse of sample trajectories across multiple parameter updates. The PPO algorithm alternates between two key phases – data collection and policy update – until a predefined number of iterations is completed. With an optimized controller that maps observed environmental observations to appropriate actuation signals, we measure the fitness of a robot by calculating the cumulative reward it receives over a complete episode, which reflects its performance in accomplishing a given task. For further details on the PPO algorithm, please refer to Schulman et al. (2017).

4 EXPERIMENTS

We begin this section with an introduction to our experimental setups, and then analyze the results of our comparison and ablation studies in detail. Our code is available on anonymous GitHub for replicability¹. Our experiments are designed to address the following questions:

- **Q1:** Can LAsER outperform state-of-the-art baselines in robot design automation?
- **Q2:** To what extent does DiRect improve the exploration-exploitation tradeoff of LLM-aided evolution?
- **Q3:** Does task metadata bring additional benefits to single-task robot design automation? Moreover, does it aid inter-task experience transfer and enable zero-shot robot design for new tasks?
- **Q4:** Previous studies have shown that different temperature parameters and versions of LLMs yield varying evolutionary outcomes. What are the specific impacts of these factors in our context?

4.1 EXPERIMENTAL SETUPS

Benchmark Setting. We base our experiments on Evolution Gym (EvoGym; Bhatia et al., 2021), a simulation environment designed for voxel-based soft robots (VSRs). In EvoGym, VSRs are represented in a grid-like layout and consist of five types of voxels: rigid voxels, soft voxels, horizontal actuators, vertical actuators, and empty voxels. VSRs achieve motion control by altering the sizes of actuators either horizontally or vertically according to action signals. For benchmarking, we select three task instances: one locomotion task, Walker-v0, which requires a robot to walk as quickly as possible on flat terrain, and two manipulation tasks, Carrier-v0 and Pusher-v0, where the robot must carry or push a rectangular object besides fast locomotion. A detailed introduction to these tasks is provided in Appendix B. For more information on EvoGym, please refer to Bhatia et al. (2021).

Baselines. We compare our method against the following baselines: (i) **Bayesian Optimization (BO; Kushner, 1964; Mockus, 1974)**, a classic algorithm designed to optimize expensive-to-evaluate functions. It employs a probabilistic model (e.g. Gaussian Process) as a surrogate for the objective function and determines where to sample based on predicted mean and uncertainty. (ii) **Speciated Evolver (SE; Medvet et al., 2021)**, a variant of the genetic algorithm (GA; Michalewicz, 2013) that divides the population into species to preserve diversity and prevent premature convergence. (iii) **RoboGAN (Hu et al., 2022)**, an estimation-of-distribution algorithm (EDA) that utilizes the Generative Adversarial Network (GAN) to track the distribution of high-performing robot designs and generate new candidate solutions. (iv) The last baseline, which we term **LLM-Tuner**, is adapted from Zhang (2024) that uses LLMs to supervise the hyperparameter tuning of a genetic algorithm. Drawing comparison with LLM-Tuner would directly verify the benefits of LLMs serving as intelligent search operators. We additionally draw comparisons with two latest baselines, with results presented in Appendix G.

¹<https://anonymous.4open.science/r/LAsER-D5C2>

Evaluation Metrics. We employ the following metrics to evaluate the performance of various approaches: **(i) Maximal Fitness**, defined as the fitness of the best-performing robot design achieved within a specific number of evaluations. This metric is commonly used in robot design automation to assess optimization efficiency. **(ii) Diversity**: Given the significance of developing diverse robotic ecosystems to handle volatile environments, we measure the diversity of high-performing robot designs² from two perspectives: one is the average edit distance among all pairs of high-performing robot designs (Saito & Oka, 2024), and the other is the total number of distinct high-performing robot designs. We further aggregate the two values via weighted averaging, where the latter is multiplied by 0.1 so that they are roughly on the same scale and given equal importance. Please refer to Appendix L for a detailed discussion on diversity measurement. We also include an analysis of computational efficiency in Appendix Q.

Implementation Details. We use GPT-4o-mini for both LAsER and LLM-Tuner, with the temperature parameter set as 0.7. For ablation studies, we additionally try out GPT-3.5-Turbo and temperatures of 1 and 1.5. Following the common practice in previous VSR studies (Song et al., 2024a; Saito & Oka, 2024; Dong et al., 2023; Bhatia et al., 2021), we choose the simple yet effective control protocol for fitness evaluation, i.e. Multilayer Perceptron (MLP) as the controller for each robot design and PPO algorithm for policy training. Following previous studies on VSR design (Song et al., 2024a; Saito & Oka, 2024; Dong et al., 2023; Bhatia et al., 2021), robot designs are constrained to a 5×5 bounding box for an expressive yet tractable search space. Nevertheless, as demonstrated in Appendix I, our approach is scalable to larger design spaces. For fair comparison, each method is permitted 1000 robot evaluations. Experimental results are averaged across three independent runs to reduce randomness (we have currently implemented two more sets of repeated experiments for LAsER and LLM-Tuner, the most competitive baseline, and the results (with significance tests) are reported in Appendix H). Our experiments are conducted on a server equipped with Intel Xeon processors running at 2.20 GHz and four NVIDIA Tesla RTX GPUs, with the system operating under Ubuntu 22.04. We relegate additional parameter settings to Appendix C. For further implementation details, please refer to our code repository.

4.2 COMPARISON STUDIES

4.2.1 SINGLE-TASK OPTIMIZATION

We begin our analysis by examining single-task optimization performances. As demonstrated in Figure 2, LAsER nearly consistently outperforms all baselines across the three tasks with significant margins. Specifically, LAsER achieves rapid convergence speeds to optimal robot designs, with only one exception on Walker-v0, where LLM-Tuner demonstrates slightly faster convergence in the early stage of evolution but ends up further from optimality. The superior performance of LAsER compared to LLM-Tuner highlights that LLMs have more important roles to play beyond merely tuning hyperparameters for traditional EAs.

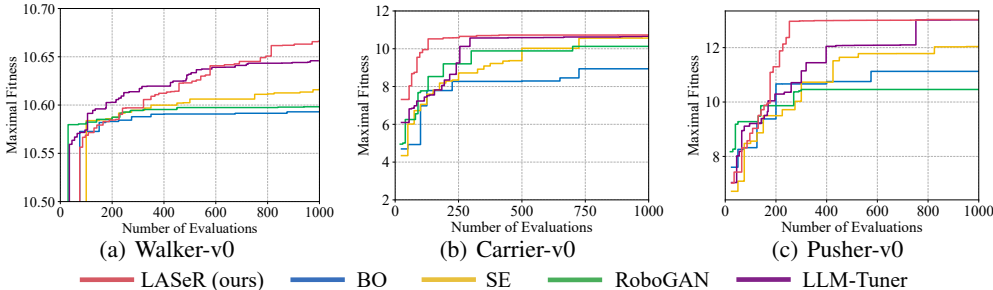


Figure 2: Comparative Results of Single-Task Optimization

Table 1 further demonstrates the diversity of high-performing robot designs achieved by different methods. We observe that LAsER surpasses all baselines without exception. For visualizations of

²Specifically, we first calculate the 90% quantile of fitnesses obtained by all methods, and consider robot designs with fitness exceeding this threshold as high-performing.

evolved robot designs please refer to Appendix K. We additionally compare the fitness performance of robot designs before and after being modified by DiRect, and find no significant difference (see Appendix F for quantitative results and examples of DiRect modifications). This suggests that our Diversity Reflection Mechanism indeed encourages the LLM to introduce variability into robot design while keeping its functionality largely intact. All these results combine to prove the distinct advantage of DiRect to promote beneficial exploratory behaviors directed towards high-performing regions. It is worth noting that the Bayesian Optimization algorithm, known for a balance between exploration and exploitation in its acquisition function, actually compromises a great deal of optimization efficiency for exploration and fails to generate high-performing robots in many cases. On the contrary, we reshape the exploration-exploitation tradeoff of LLM-aided evolution to yield dual benefits in optimization efficiency and diversity. We performed additional comparison on Catcher-v0, one of the most challenging tasks in EvoGym, to further showcase the effectiveness of LAsER (see Appendix E).

Table 1: Comparative Results of Diversity

	Walker-v0	Carrier-v0	Pusher-v0
BO	N/A	8.35 (N/A)	11.31 (N/A)
SE	5.40 (0.30)	15.84 (0.77)	6.61 (3.45)
RoboGAN	N/A	10.94 (N/A)	N/A
LLM-Tuner	19.21 (11.25)	20.33 (6.53)	16.75 (5.95)
LAsER(ours)	20.77 (4.77)	22.11 (3.88)	27.56 (1.66)

Note 1: Since we have three repeated experiments, the results are reported as “mean(standard deviation)”. The same is true for Table 2.

Note 2: When no more than one high-performing robot design is produced, diversity cannot be calculated. When this is the case across all repeated experiments (e.g. BO on Walker-v0), the cell is filled with “N/A”. If this is the case for two repeated experiments (e.g. BO on Carrier-v0), the standard deviation is unavailable and only the mean is reported.

Note 3: For separate results of edit distance and the number of high-performing designs, please refer to Appendix L.

4.2.2 INTER-TASK KNOWLEDGE TRANSFER

Now we proceed to explore the ability of LLMs to transfer design experience across different tasks. To achieve this purpose, we introduce two more tasks: BridgeWalker-v0 and UpStepper-v0. Specifically, both BridgeWalker-v0 and UpStepper-v0 bear some resemblance to Walker-v0, but differ in their terrains: BridgeWalker-v0 involves locomotion on a soft rope-bridge, whereas UpStepper-v0 requires climbing stairs of varying lengths. The LLM is prompted to generate robot designs for each new task, given elite Walker-v0 robot designs. As shown in Figure 3(b)³, the zero-shot proposals by LLM outperform both randomly generated designs and those provided elite Walker-v0 designs which have finished their evolving with LAsER, in terms of accomplishing the new tasks. This serves as sound evidence that the LLM is not simply replicating exemplars in its context, but rather assimilating design experience that is beneficial for new settings. This is largely owing to our incorporation of task-related metadata that provokes inter-task reasoning within the LLM. For illustration, Figure 3(a) demonstrates some insights that the LLM drew from Walker-v0 elites to transfer to BridgeWalker-v0.

The zero-shot proposals are then leveraged as the initial population for further optimization. Figure 3(c-1) and 3(c-2) demonstrate that this informative initialization results in faster evolution than starting from scratch, and pulls away from baseline algorithms with even greater advantage. Also note that the zero-shot proposals for BridgeWalker-v0 turn out to be already near optimal before undergoing marginal improvement with evolution. These promising results unprecedentedly uncover the possibility of generalizable evolutionary processes driven by LLMs and hopefully inspire closer investigation in future work. Please note that while here we focused on intuitively similar task instances, we prove in Appendix O that this prior knowledge regarding inter-task relationships is not necessary for successful experience transfer.

³The result is averaged over ten robot designs in each case.

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

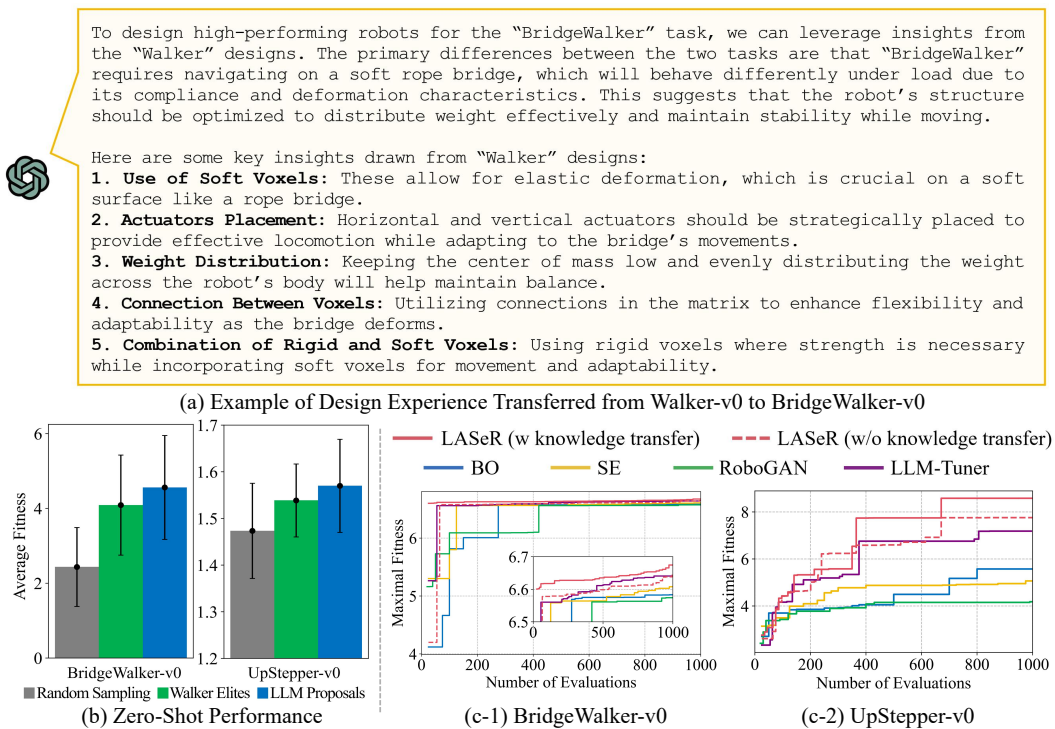


Figure 3: Effectiveness of Inter-Task Knowledge Transfer

4.3 ABLATION STUDIES

4.3.1 EFFECTIVENESS OF DiRECT

As shown in Table 2, the Diversity Reflection Mechanism fosters a robust increase in diversity compared to an ablated version. It is further demonstrated in Figure 4 that the exploratory behaviors led by DiRect also facilitate more efficient navigating of design spaces, leading to reduced susceptibility to local optima and higher optimization efficiency. These results combine to underscore the distinct superiority of DiRect to yield dual benefits in optimization efficiency and diversity by exploiting the reasoning capabilities of LLMs.

Table 2: Ablative Results of Diversity

	Walker-v0	Carrier-v0	Pusher-v0
LLaSeR	20.77 (4.77)	22.11 (3.88)	27.56 (1.66)
LLaSeR w/o DiRect	16.96 (1.38)	7.28 (1.13)	10.50 (1.04)

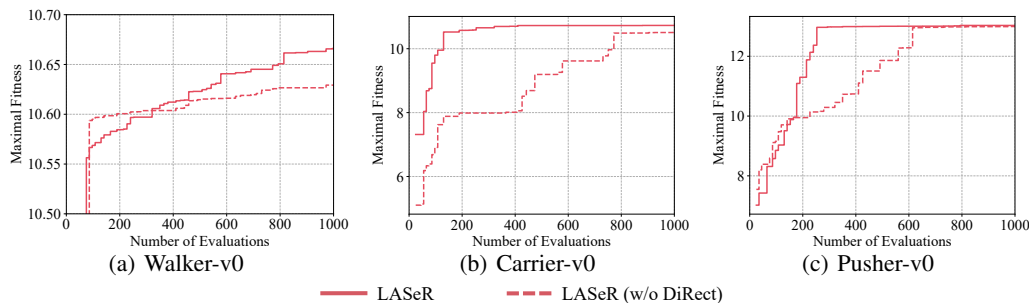


Figure 4: Effectiveness of DiRect

4.3.2 EFFECTIVENESS OF TASK-RELATED METADATA

As explained in Section 3.2, current LLM-aided evolutionary frameworks mostly lack sufficient grounding in task-related background information, which potentially impedes their performances in real-world applications. We test this conjecture by removing descriptions of task objectives and simulation environment from our prompts, and see a significant performance drop (Figure 5(a)), hence justifying our prompt design. [For finer-grained ablations on individual components of our prompt, please see Appendix M.](#)

4.3.3 IMPACT OF LLM VERSION AND TEMPERATURE PARAMETER

Previous work has shown that the temperature parameter of LLMs has an unignorable influence on evolutionary outcomes, with higher temperatures tending to yield better results (Pluhacek et al., 2024). However, we observe a reverse effect where a lower temperature turns out slightly more favorable (Figure 5(b)). We suspect that this is partly due to the complexity within VSR design, which necessitates precise extrapolation from an ascending sequence of solutions. Any deviation could lead to substantial performance drops, outweighing the benefits of random exploration. This again underscores the superiority of our proposed DiRect mechanism, which resorts to more educated exploration strategies. Meanwhile, we note that the ablation studies with temperature as 1.5 fail similarity checks only about 70% as often as when temperature equals 0.7. In words, higher temperatures would lead to greater but ineffective variability in candidate solutions so that they could bypass diversity reflection. These results suggest that lower output temperatures are required for our approach to work better. Additionally, we observe the same improvement resulted from more up-to-date LLMs as in past literature (Figure 5(c)). This shows promise of robot design automation directly benefiting from better language models, which puts us in a strategic position to ride the wave of rapidly progressing LLMs.

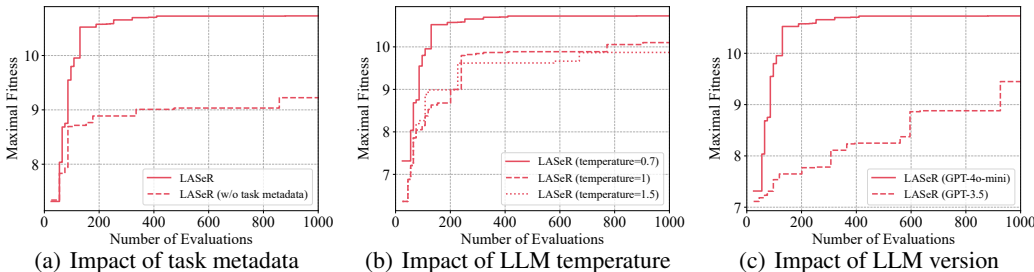


Figure 5: Additional ablation studies on Carrier-v0

5 CONCLUDING REMARKS

We present LLaSeR – Large Language Model-Aided Evolutionary Search for Robot Design Automation. With a novel diversity-oriented reflection mechanism termed DiRect, we elicit intelligent exploratory behaviors from LLMs that reshape the exploration-exploitation tradeoff with dual improvements in optimization efficiency and diversity. We additionally propose to ground robot design on rich task-related metadata and uncover the intriguing inter-task reasoning capabilities of LLMs to foster generalizable design processes across different applications. Our experiments with simulated voxel-based soft robots demonstrate superior performances of our approach compared to competitive baselines. Scaling up LLaSeR for multi-task optimization would hopefully further harness the inter-task reasoning abilities of LLMs to boost sample efficiency. The recent advancements in prompt engineering, such as Chain-of-Thoughts (Wei et al., 2022) and Tree-of-Thoughts (Yao et al., 2024), also hold promise for further unleashing the potential of LLMs in robot design automation and warrant further examination. Moreover, it is interesting to investigate how LLM-aided control strategies (Wang et al., 2023a) could be integrated into our framework, so that LLMs are not only responsible for action planning, but also enabled to design their own embodiments, hence exploiting the synergy between design and control. We leave these for our future work. [For more detailed discussions of limitations and open problems for future research, please see Appendix P.](#)

540 ETHICS STATEMENT

541

542 This work uses simulated task environments which have been commonly used in previous research
 543 of robot design automation and should not be regarded controversial. Our use of Large Language
 544 Models is strictly confined to simulated robot design generation without real-world deployment, and
 545 therefore does not involve any safety risks.

546

547 REPRODUCIBILITY STATEMENT

548

549 Our code is readily available on anonymous GitHub. This work uses GPT-3-turbo and GPT-4o-
 550 mini, whose APIs are publicly accessible. However, due to the uncontrollable random generator
 551 seeds behind close-source LLMs, experiments involving these models generally suffer from poor
 552 reproducibility (Huang et al., 2024a). Developing reproducible methods for API calls would signif-
 553 icantly improve the replicability of research outcomes involving Large Language Models.

554 REFERENCES

555

556 Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Ale-
 557 man, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical
 558 report. *arXiv preprint arXiv:2303.08774*, 2023.

559

560 Jagdeep Bhatia, Holly Jackson, Yunsheng Tian, Jie Xu, and Wojciech Matusik. Evolution gym:
 561 A large-scale benchmark for evolving soft robots. *Advances in Neural Information Processing*
 562 *Systems*, 34:2201–2214, 2021.

563 Shuvayan Brahmachary, Subodh M Joshi, Aniruddha Panda, Kaushik Koneripalli, Arun Ku-
 564 mar Sagotra, Harshil Patel, Ankush Sharma, Ameya D Jagtap, and Kaushic Kalyanaraman.
 565 Large language model-based evolutionary optimizer: Reasoning with elitism. *arXiv preprint*
 566 *arXiv:2403.02054*, 2024.

567

568 Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choroman-
 569 ski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action
 570 models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

571 Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao
 572 Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-
 573 scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

574

575 Nick Cheney, Robert MacCurdy, Jeff Clune, and Hod Lipson. Unshackling evolution: evolving soft
 576 robots with multiple materials and a powerful generative encoding. *ACM SIGEVOlution*, 7(1):
 577 11–23, 2014.

578 Olivier Chocron and Philippe Bidaud. Evolutionary algorithms in kinematic design of robotic sys-
 579 tems. In *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and*
 580 *Systems. Innovative Robotics for Real-World Applications. IROS'97*, volume 2, pp. 1111–1117.
 581 IEEE, 1997.

582 François Cochevelou, David Bonner, and Martin-Pierre Schmidt. Differentiable soft-robot gener-
 583 ation. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 129–137,
 584 2023.

585

586 Heng Dong, Junyu Zhang, and Chongjie Zhang. Leveraging hyperbolic embeddings for coarse-to-
 587 fine robot design. *arXiv preprint arXiv:2311.00462*, 2023.

588

589 Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair,
 590 Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the*
 591 *ACM*, 63(11):139–144, 2020.

592 Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian,
 593 and Yujiu Yang. Connecting large language models with evolutionary algorithms yields powerful
 prompt optimizers. *arXiv preprint arXiv:2309.08532*, 2023.

- 594 Jonathan Hiller and Hod Lipson. Automatic design and manufacture of soft robots. *IEEE Transactions on Robotics*, 28(2):457–466, 2011.
- 595
- 596
- 597 Jiaheng Hu, Julian Whitman, Matthew Travers, and Howie Choset. Modular robot design optimization with generative adversarial networks. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 4282–4288. IEEE, 2022.
- 598
- 599
- 600 Jiaheng Hu, Julian Whitman, and Howie Choset. Glso: Grammar-guided latent space optimization for sample-efficient robot design automation. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 1321–1331. PMLR, 14–18 Dec 2023. URL <https://proceedings.mlr.press/v205/hu23c.html>.
- 601
- 602
- 603
- 604
- 605
- 606 Beichen Huang, Xingyu Wu, Yu Zhou, Jibin Wu, Liang Feng, Ran Cheng, and Kay Chen Tan. Exploring the true potential: Evaluating the black-box optimization capability of large language models. *arXiv preprint arXiv:2404.06290*, 2024a.
- 607
- 608
- 609 Sen Huang, Kaixiang Yang, Sheng Qi, and Rui Wang. When large language model meets optimization. *arXiv preprint arXiv:2405.10098*, 2024b.
- 610
- 611
- 612 Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- 613
- 614 Sam Kriegman, Douglas Blackiston, Michael Levin, and Josh Bongard. A scalable pipeline for designing reconfigurable organisms. *Proceedings of the National Academy of Sciences*, 117(4):1853–1859, 2020a.
- 615
- 616
- 617 Sam Kriegman, Amir Mohammadi Nasab, Dylan Shah, Hannah Steele, Gabrielle Branin, Michael Levin, Josh Bongard, and Rebecca Kramer-Bottiglio. Scalable sim-to-real transfer of soft robot designs. In *2020 3rd IEEE international conference on soft robotics (RoboSoft)*, pp. 359–366. IEEE, 2020b.
- 618
- 619
- 620
- 621 Sam Kriegman, Douglas Blackiston, Michael Levin, and Josh Bongard. Kinematic self-replication in reconfigurable organisms. *Proceedings of the National Academy of Sciences*, 118(49):e2112672118, 2021.
- 622
- 623
- 624
- 625 Harold J Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. 1964.
- 626
- 627
- 628 Robert Lange, Yingtao Tian, and Yujin Tang. Large language models as evolution strategies. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 579–582, 2024.
- 629
- 630
- 631 Chris Leger. *Darwin2K: An evolutionary approach to automated design for robotics*, volume 574. Springer Science & Business Media, 2012.
- 632
- 633
- 634 Julie Legrand, Seppe Terry, Ellen Roels, and Bram Vanderborght. Reconfigurable, multi-material, voxel-based soft robots. *IEEE Robotics and Automation Letters*, 8(3):1255–1262, 2023.
- 635
- 636
- 637 Joel Lehman, Jonathan Gordon, Shawn Jain, Kamal Ndousse, Cathy Yeh, and Kenneth O Stanley. Evolution through large models. In *Handbook of Evolutionary Machine Learning*, pp. 331–366. Springer, 2023.
- 638
- 639
- 640 Bryan Lim, Manon Flageat, and Antoine Cully. Large language models as in-context ai generators for quality-diversity. *arXiv preprint arXiv:2404.15794*, 2024.
- 641
- 642
- 643 Shengcai Liu, Caishun Chen, Xinghua Qu, Ke Tang, and Yew-Soon Ong. Large language models as evolutionary optimizers. In *2024 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8. IEEE, 2024a.
- 644
- 645
- 646 Tennison Liu, Nicolás Astorga, Nabeel Seedat, and Mihaela van der Schaar. Large language models to enhance bayesian optimization. In *The Twelfth International Conference on Learning Representations*, 2024b. URL <https://openreview.net/forum?id=00xotBmGol>.
- 647

- 648 Yang Liu, Weixing Chen, Yongjie Bai, Jingzhou Luo, Xinshuai Song, Kaixuan Jiang, Zhida Li,
649 Ganlong Zhao, Junyi Lin, Guanbin Li, et al. Aligning cyber space with physical world: A com-
650 prehensive survey on embodied ai. *arXiv preprint arXiv:2407.06886*, 2024c.
- 651
- 652 Pingchuan Ma, Tsun-Hsuan Wang, Minghao Guo, Zhiqing Sun, Joshua B Tenenbaum, Daniela Rus,
653 Chuang Gan, and Wojciech Matusik. Llm and simulation as bilevel optimizers: A new paradigm
654 to advance physical scientific discovery. *arXiv preprint arXiv:2405.09783*, 2024.
- 655 Eric Medvet, Alberto Bartoli, Federico Pigozzi, and Marco Rochelli. Biodiversity in evolved voxel-
656 based soft robots. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp.
657 129–137, 2021.
- 658
- 659 Elliot Meyerson, Mark J Nelson, Herbie Bradley, Adam Gaier, Arash Moradi, Amy K Hoover, and
660 Joel Lehman. Language model crossover: Variation through few-shot prompting. *arXiv preprint*
661 *arXiv:2302.12170*, 2023.
- 662 Zbigniew Michalewicz. *Genetic algorithms+ data structures= evolution programs*. Springer Sci-
663 ence & Business Media, 2013.
- 664
- 665 Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Are-
666 nas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. Large language models as general pattern
667 machines. *arXiv preprint arXiv:2307.04721*, 2023.
- 668 Jonas Mockus. On bayesian methods for seeking the extremum. In *Proceedings of the IFIP Techni-
669 cal Conference*, pp. 400–404, 1974.
- 670
- 671 Clint Morris, Michael Jurado, and Jason Zutty. Llm guided evolution-the automation of models
672 advancing models. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp.
673 377–384, 2024.
- 674 Michal Pluhacek, Jozef Kovac, Peter Janku, Tomas Kadavy, Roman Senkerik, and Adam Viktorin.
675 A critical examination of large language model capabilities in iteratively refining differential evo-
676 lution algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference Com-
677 panion*, pp. 1855–1862, 2024.
- 678 Kevin Qiu, Krzysztof Ciebiera, Paweł Fijałkowski, Marek Cygan, and Łukasz Kuciński.
679 Robomorph: Evolving robot morphology using large language models. *arXiv preprint*
680 *arXiv:2407.08626*, 2024.
- 681
- 682 Bernardino Romera-Paredes, Mohammadamin Barekatin, Alexander Novikov, Matej Balog,
683 M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang,
684 Omar Fawzi, et al. Mathematical discoveries from program search with large language models.
685 *Nature*, 625(7995):468–475, 2024.
- 686 Nicholas Roy, Ingmar Posner, Tim Barfoot, Philippe Beaudoin, Yoshua Bengio, Jeannette Bohg,
687 Oliver Brock, Isabelle Depatie, Dieter Fox, Dan Koditschek, et al. From machine learn-
688 ing to robotics: Challenges and opportunities for embodied intelligence. *arXiv preprint*
689 *arXiv:2110.15245*, 2021.
- 690
- 691 Takumi Saito and Mizuki Oka. Effective design and interpretation in voxel-based soft robotics: A
692 part assembly approach with bayesian optimization. In *Artificial Life Conference Proceedings*
693 36, volume 2024, pp. 26. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA
694 journals-info . . . , 2024.
- 695 John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy
696 optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- 697
- 698 Dylan S Shah, Joshua P Powers, Liana G Tilton, Sam Kriegman, Josh Bongard, and Rebecca
699 Kramer-Bottiglio. A soft robot that adapts to environments through shape change. *Nature Ma-
700 chine Intelligence*, 3(1):51–59, 2021.
- 701 Karl Sims. Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer*
graphics and interactive techniques, pp. 15–22, 1994.

- 702 Junru Song, Yang Yang, Wei Peng, Weien Zhou, Feifei Wang, and Wen Yao. Morphvae: Advancing
703 morphological design of voxel-based soft robots with variational autoencoders. In *Proceedings*
704 *of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 10368–10376, 2024a.
- 705
706 Xingyou Song, Yingtao Tian, Robert Tjarko Lange, Chansoo Lee, Yujin Tang, and Yutian Chen.
707 Position: Leverage foundational models for black-box optimization. In *Forty-first International*
708 *Conference on Machine Learning*, 2024b.
- 709 Francesco Stella, Cosimo Della Santina, and Josie Hughes. How can llms transform the robotic
710 design process? *Nature machine intelligence*, 5(6):561–564, 2023.
- 711
712 Luke Strgar, David Matthews, Tyler Hummer, and Sam Kriegman. Evolution and learning in differ-
713 entiable robots. *arXiv preprint arXiv:2405.14712*, 2024.
- 714 Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu,
715 Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly
716 capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- 717 InternLM Team. Internlm: A multilingual language model with progressively enhanced capabilities,
718 2023.
- 719
720 Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Niko-
721 lay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open founda-
722 tion and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- 723
724 Thanh VT Tran and Truong Son Hy. Protein design by directed evolution guided by large language
725 models. *IEEE Transactions on Evolutionary Computation*, 2024.
- 726 Tingwu Wang, Yuhao Zhou, Sanja Fidler, and Jimmy Ba. Neural graph evolution: Towards efficient
727 automatic robot design. *arXiv preprint arXiv:1906.05370*, 2019.
- 728
729 Yen-Jen Wang, Bike Zhang, Jianyu Chen, and Koushil Sreenath. Prompt a robot to walk with large
730 language models. *arXiv preprint arXiv:2309.09969*, 2023a.
- 731 Yuxing Wang, Shuang Wu, Tiantian Zhang, Yongzhe Chang, Haobo Fu, Qiang Fu, and Xueqian
732 Wang. Preco: Enhancing generalization in co-design of modular soft robots via brain-body pre-
733 training. In *Conference on Robot Learning*, pp. 478–498. PMLR, 2023b.
- 734 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny
735 Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in*
736 *neural information processing systems*, 35:24824–24837, 2022.
- 737
738 Xingyu Wu, Sheng-hao Wu, Jibin Wu, Liang Feng, and Kay Chen Tan. Evolutionary computation
739 in the era of large language model: Survey and roadmap. *arXiv preprint arXiv:2401.10034*, 2024.
- 740 Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun
741 Chen. Large language models as optimizers, 2024. URL <https://arxiv.org/abs/2309.03409>.
- 742
743 Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik
744 Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Ad-*
745 *vances in Neural Information Processing Systems*, 36, 2024.
- 746
747 Haoran Ye, Jiarui Wang, Zhiguang Cao, Federico Berto, Chuanbo Hua, Haeyeon Kim, Jinkyoo Park,
748 and Guojie Song. Large language models as hyper-heuristics for combinatorial optimization,
749 2024. URL <https://arxiv.org/abs/2402.01145>.
- 750 Lechen Zhang. Cuda-accelerated soft robot neural evolution with large language model supervision.
751 *arXiv preprint arXiv:2405.00698*, 2024.
- 752
753
754
755

756 APPENDIX

757
758 A FULL PROMPTS

759
760 A.1 SYSTEM PROMPT FOR GENERATING OFFSPRING ROBOT DESIGNS

761
762 Now you will serve as an intelligent search operator in an Evolutionary Algorithm. In each generation
763 you are given a number of evaluated solutions in the format of numpy array, together with their fitness
764 scores. Each solution and its fitness score are separated by a comma. Different solutions are separated
765 by semicolons. The solutions are sorted according to their fitness scores in ascending order. Higher
766 fitness scores are better. Your job is to output a new solution that meets a desired fitness. Please
767 try your best to logically analyze the relationship between the evaluated solutions and their fitness
768 scores, and adhere to this information while proposing the new solution. A solution is a 5 times 5 matrix,
769 where each entry is an integer between zero and four. Please begin the new solution with <solution>
770 and end it with </solution>. The new solution should be formatted in numpy array fashion. The new solution
771 must be distinct from the evaluated solutions. Only generate the new solution. No explanation.

772
773 A.2 USER PROMPT FOR GENERATING OFFSPRING ROBOT DESIGNS (CARRIER-V0 AS AN
774 EXAMPLE)

775
776 **## Description of VSRs**

777 We are going to design the structure of a two-dimensional voxel-based soft robot (VSR) in a simulation
778 environment. VSRs are composed of square-shaped voxels of different types, aligned into a 5 * 5 matrix.
779 Adjacent voxels (that is, in either the same row or the same column) are connected together; Voxels
780 located in diagonal positions are not connected together. The robot is subject to gravity, and the bottom
781 row touches the ground. There are 5 types of voxels available, including soft voxels (for which elastic
782 deformation is possible), rigid voxels (which can not deform), horizontal and vertical actuators (which
783 can change their sizes horizontally or vertically), and empty voxels (which basically mean that the
784 corresponding position is empty). Empty voxels, rigid voxels, soft voxels, horizontal actuators and
785 vertical actuators are represented as 0, 1, 2, 3 and 4, respectively.

786 **## Description of simulation environment**

787 The simulation represents objects and their environment as a 2D mass-spring system in a grid-like layout,
788 where objects are initialized as a set of non-overlapping, connected voxels. The simulation converts
789 all objects into a set of point masses and springs by turning each voxel into a cross-braced square,
790 which may undergo deformation as the simulation progresses. The springs obey Hooke's law. Note that
791 adjacent voxels share point masses and springs on their common edge. All point masses in the simulation
792 have the same mass and the equilibrium lengths of axis-aligned and diagonal springs are constants for
793 simplicity. However, the spring constants assigned vary based on voxel material-type, with ties broken
794 in favor of the more-rigid spring. The actuators undergo gradual expansion/contraction either
795 horizontally or vertically according to action signals, by changing the lengths of the corresponding
796 springs.

797 **## Task description**

798 Your job is to propose robot designs suitable for completing the following task. A three-voxel wide
799 box is initialized right above the robot, and the robot is required to keep the box on top of its head
800 stably without letting it slip off, while locomoting rightwards as quickly as possible.

801 **## Constraints**

802 There are two constraints to VSR designs: 1. all voxels must form an entirety and should not fall apart;
803 That is, the four voxels, if any, above, below, to the left and to the right of a non-empty voxel mustn't
804 be empty at the same time. An example that violates such a constraint is
805 $[[2,2,2,2,2],[1,0,1,0,1],[0,4,3,4,0],[1,3,1,0,1],[0,4,2,4,0]]$, in which the voxel '1' in the fourth
806 row and fifth column would fall off because it is not connected to any non-empty voxel; (2) there must
807 be at least one actuator (that is, either 3 or 4), so that the robot could interact with the environment.
808
809

810 **## Additional requirements**
 811 Please carefully analyze the relationship between evaluated solutions and their fitness scores, and
 812 make use of this information to propose the new solution. Please make use of empty voxels cleverly so
 813 that complex functional substructures could be produced to fulfill the purposes of both carrying and
 814 locomoting. Note that a high-performing robot design is not necessarily symmetric.

815 (elite design-fitness pairs omitted, sorted in ascending order)

816

817 **## "Just-ask" query**
 818 Now please generate a new robot design that has a fitness of {str(1.2*current_max_fitness)}.

821 A.3 USER PROMPT OF DIRECT

822

823 The solution that you just generated is too similar with an existing one. It needs further modification
 824 to improve diversity. Please decide which voxels in the solution can be replaced by other types of materials,
 825 without harming its fitness score. Change no more than 3 voxels. Please base your analysis on the
 826 characteristics of the evaluated solutions given to you. Meanwhile, make sure that the modification
 827 does not violate the constraints of VSRs. Now please tell me which voxels exactly do you think can be
 828 altered, and explain the reason.

829 (LLM suggesting modifications)

830

831 Based on your analysis above, please generate the resulting solution. The number of voxels changed should
 832 not exceed three. Do not provide further textual explanation.

835 A.4 USER PROMPT FOR INTER-TASK KNOWLEDGE TRANSFER (WALKER-V0 → 836 BRIDGEWALKER-V0 AS AN EXAMPLE)

837

838 (Descriptions of VSRs, simulation environment and constraints are same as those in A.1 and therefore
 839 omitted.)

840

841 We already have some high-performing robot designs from a task named 'Walker', where the robot is required
 842 to locomote rightwards as quickly as possible on rigid flat terrain. The robot designs are as follows:

843 (Elite Walker-v0 designs omitted.)

844

845 Now your job is to propose ten high-performing robot designs for another task named 'BridgeWalker',
 846 where the robot is required to locomote rightwards as quickly as possible on a soft rope-bridge rather
 847 than rigid flat terrain. Please analyze the potential correlation between the two tasks, identify the
 848 knowledge that can be transferred from Walker to BridgeWalker, and give ten robot designs that are suitable
 849 for BridgeWalker. Give your answer in numpy array fashion. Enclose each design between <Solution> and
 850 </Solution>.

852 B INTRODUCTION TO TASK INSTANCES

853 In this section, we briefly introduce the tasks we adopted for benchmarking in Evolution Gym. This
 854 introduction is heavily borrowed from their original paper (Bhatia et al. 2021).

855 We first define some notations that would be used later.

- 856 • **Position:** Denote with p^o the position of the center of mass of an object o , which consists
 857 of two components p_x^o and p_y^o , i.e. the positions on x and y axis. p^o is derived by averaging
 858 the positions of all the point-masses that make up object o ;
- 859 • **Velocity:** Denote with v^o the velocities of the center of mass of an object o , which consists
 860 of two components v_x^o and v_y^o , i.e. the velocity on x and y axis. v^o is computed by averaging
 861 the velocities of all point masses that make up object o ;

- **Orientation:** Denote with θ^o , a vector of length one, the orientation of an object o . Denote the position of point mass i of object o as p_i , and θ^o is computed by averaging over all i the angle between the vector $p_i - p^o$ at current time and the initial state. This average is weighted by $\|p_i - p^o\|$ in the initial state.
- **Other observations:** Let c^o be a vector of length $2n$ that describes the relative positions of all n point masses of object o to the center of mass. Let $h_b^o(d)$ characterize the terrain information around a robot below its center of mass. More specifically, for some integer $x \leq d$, the corresponding entry in vector $h_b^o(d)$ will be the highest point of the terrain which is lower than p_y^o between a range of $[x, x + 1]$ voxels from p_x^o in the x -direction.
- Besides, we would denote the robot as object r , the box that it is trying to manipulate as object b , the number of point masses in r as n , the observation vector as \mathcal{S} , and the reward function as R .

B.1 WALKER-v0



Figure 6: Walker-v0

In this task, the robot is required to walk as far as possible on flat terrain. $\mathcal{S} \in \mathbb{R}^{n+2}$ consists of v^r and c^r with lengths 2 and n . $R = \Delta p_x^r$ rewards the robot for moving in the positive x -direction. The robot is also given a one-time reward of 1 for reaching the end of the terrain.

B.2 CARRIER-v0



Figure 7: Carrier-v0

In this task, the robot is required to catch a box initialized above it and carries it as far as possible. $\mathcal{S} \in \mathbb{R}^{n+6}$ consists of v^b , $p^b - p^r$, v^r and c^r with lengths 2, n , 2 and 2 respectively. $R = R_1 + R_2$, where $R_1 = 0.5 \cdot \Delta p_x^r + 0.5 \cdot \Delta p_x^b$ rewards the robot and the box for moving in the positive x -direction, and $R_2 = 0$ if $p_y^b \geq t_y$ and otherwise $10 \cdot \Delta p_y^b$ penalizes the robot for dropping the box below a threshold height t_y . The robot is also given a one-time reward of 1 for reaching the end of the terrain.

B.3 PUSHER-v0



Figure 8: Pusher-v0

In this task, the robot is required to push a box initialized in front of it. $\mathcal{S} \in \mathbb{R}^{n+6}$ consists of v^b , $p^b - p^r$, v^r and c^r with lengths 2, n , 2 and 2 respectively. $R = R_1 + R_2$, where $R_1 = 0.5 \cdot \Delta p_x^r + 0.75 \cdot \Delta p_x^b$ rewards the robot and the box for moving in the positive x -direction, and $R_2 = -\Delta |p_x^b - p_x^r|$ penalizes the robot and the box for separating in the x -direction. The robot is also given a one-time reward of 1 for reaching the end of the terrain.

B.4 BRIDGEWALKER-v0

In this task, the robot is required to walk as far as possible on a soft rope-bridge. $\mathcal{S} \in \mathbb{R}^{n+3}$ consists of v^r , θ^r and c^r with lengths 2, 1 and n respectively. $R = \Delta p_x^r$ rewards the robot for moving in

918
919
920
921
922
923
924
925
926
927



Figure 9: BridgeWalker-v0

the positive x -direction. The robot is also given a one-time reward of 1 for reaching the end of the terrain.

928
929
930
931
932
933
934

B.5 UPSTEPPER-V0

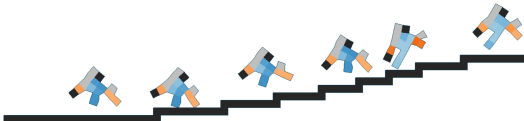


Figure 10: UpStepper-v0

935
936
937
938
939
940
941

In this task, the robot is required to mount stairs of varying lengths. $\mathcal{S} \in \mathbb{R}^{n+14}$ consists of v^r , θ^r , c^r and $h_b^r(5)$ with lengths 2, 1, n and 11, respectively. $R = \Delta p_x^r$ rewards the robot for moving in the positive x -direction. The robot is given a one-time reward of 2 for reaching the end of the terrain, and a one-time penalty of -3 for rotating more than 75 degrees from its original orientation in either direction (after which the environment is reset).

942
943

C HYPERPARAMETER SETTINGS

944
945
946
947
948
949
950
951
952
953
954

Table 3: Hyperparameter settings

hyperparameter	value
LASeR	
robot size (height \times width)	5×5
upper limit of LLM interactions	200
probability of similarity check in DiRect	0.4
similarity threshold in DiRect	no more than 20 identical voxels
percentage of survivors in natural selection	linearly decreasing from 60% to 8%
PPO Policy Training	
number of parallel sampling processes	4
number of time steps in each process	128
learning rate	2.5×10^{-4}
ϵ in the clip function of PPO	0.1
number of iterations	1000
number of epochs per iteration	4
number of mini-batches per epoch	4
λ in generalized advantage estimation (GAE)	0.95

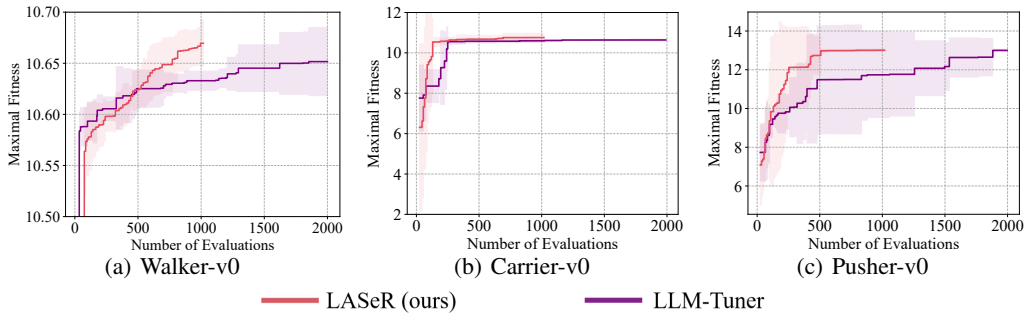
955
956
957
958
959
960
961
962
963
964

D PROLONGED EXPERIMENTS OF LLM-TUNER

965
966
967
968
969
970
971

We re-implemented LLM-Tuner, the most competitive baseline, for 2000 robot evaluations, which is double the number we employed in our original experiments. These experiments are intended to further verify that our rapid convergence is neither due to local optima or unreasonably easy task settings. Notably, as showcased in Figure 11, LLM-Tuner does not end up with higher fitness levels than those achieved by LASeR, largely confirming that our algorithm has not been stuck in

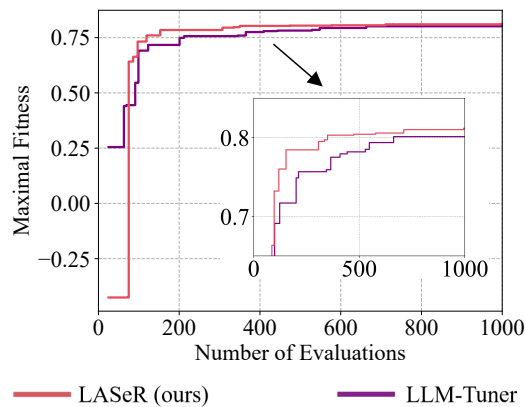
972 local optima. Meanwhile, the evidently slower convergence of LLM-Tuner, which is especially
 973 pronounced in Walker-v0 and Pusher-v0, indicates that our fast convergence is more likely due to
 974 the effectiveness afforded by LLM-aided evolution and diversity reflection mechanism, rather than
 975 an artifact of task difficulty.
 976



988 Figure 11: LLaSeR (red) compared with LLM-Tuner (purple), when the latter is implemented for
 989 2000 robot evaluations. The colored bands represent $\text{mean} \pm 1.96 \times \text{standard deviation}$. The results
 990 of LLM-Tuner are averaged across 3 independent runs, while those of LLaSeR are averaged over 5
 991 runs.
 992
 993

994 E EVALUATION ON CATCHER-V0

995
 996 We additionally performed comparison between LLaSeR and LLM-Tuner (the most competitive
 997 baseline algorithm) on Catcher-v0, one of the most challenging task instances in EvoGym where
 998 the robot is required to catch a fast-moving, rotating box. As shown in Figure 12, the advantage
 999 of LLaSeR remains evident. Notably, despite starting with inferior initialization, LLaSeR is able to
 1000 swiftly catch up with and surpass LLM-Tuner. We plan to continue with our evaluations on more
 1001 complex tasks to fully demonstrate the effectiveness of LLaSeR.
 1002



1017 Figure 12: Comparison of LLaSeR (red) and LLM-Tuner (purple) on Catcher-v0. The experimental
 1018 results are averaged across three independent runs.
 1019
 1020

1021 F COMPARISON WITH RANDOM VOXEL EDITING

1022
 1023 To further showcase the effectiveness of our diversity reflection mechanism (DiRect), we re-
 1024 implemented our experiments with DiRect replaced by random voxel mutations. Specifically, we
 1025 implemented random editing by substituting the diversity reflection (DiRect) mechanism with ran-
 dom voxel mutation, which is supported by a built-in function of EvoGym. Specifically, we found

that the number of voxels edited by DiRect in each design is about **2.61** on average. Thus, for random mutation, we set the mutation rate to be **0.1**, i.e. each voxel will, with a probability of 0.1, be randomly replaced by a different material. Given that a robot design consists of 25 voxels, this results in **2.5** voxels being edited on average, which we believe is reasonably close to DiRect editing.

We conducted a paired two-tailed Student’s t -test, and found that the fitnesses of randomly mutated robot designs are significantly lower than their pre-editing counterparts ($p < 0.001$). In contrast, the fitnesses of robot designs before and after DiRect modification show no significant difference ($p = 0.19$). The evolution with random editing also suffers from reduced optimization efficiency (Figure 13), as the stochastic exploratory behavior often disrupts essential functional structures. LLM-aided diversity reflection, on the other hand, holds a distinct “*informed*” nature. Specifically, it builds on successful designs discovered along the evolutionary trajectory, and promotes exploration without compromising functionality.

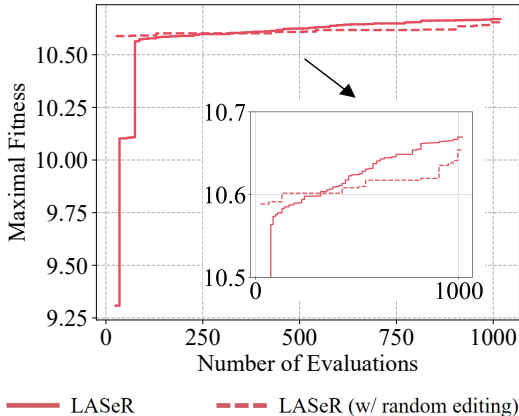


Figure 13: Comparison between diversity reflection (red solid line) and random editing (red dashed line). It is worth noting that LASeR w/ DiRect was able to swiftly catch up with and surpass LASeR w/ random editing, despite the inferior initialization.

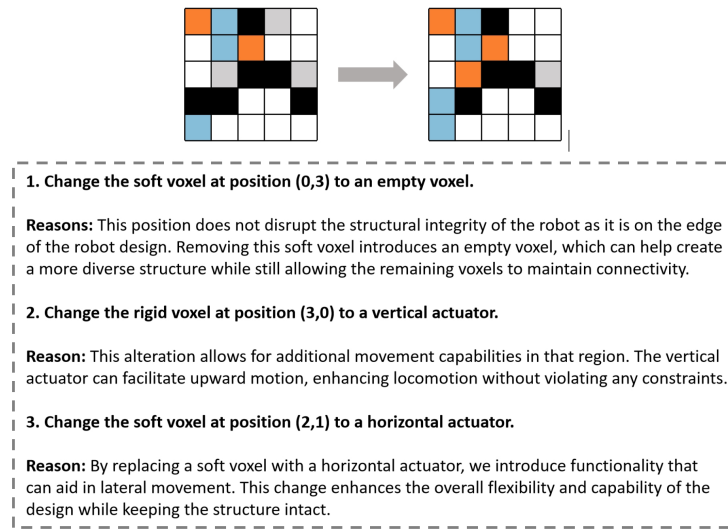
For illustrative purposes, here we present two examples of diversity reflection, including both the pre- and post-editing morphologies, along with explanations provided by the LLM for its modifications. It can be seen from Figure 14 that the LLM is indeed able to identify critical substructures within robot designs, and modifies only the voxel placements that do not affect functionality, yet promote diversity. These results provide sound evidence that DiRect is reliably functioning as an *intelligent mutation operator*.

G COMPARISON WITH TWO ADDITIONAL BASELINES

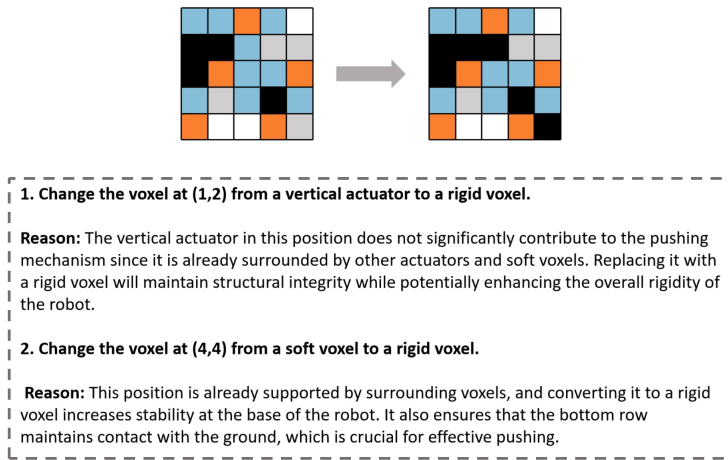
In response to the reviewers’ suggestions, we conducted comparative studies between LASeR and two additional baseline algorithms. The first one is OPRO (Yang et al., 2024), another evolutionary strategy that uses LLMs as search operators, which we adapted for voxel-based soft robot (VSR) design. The second one is MorphVAE (Song et al., 2024a), a state-of-the-art co-design algorithm that does not employ LLMs but is also developed on the EvoGym platform.

As shown in Figure 15, LASeR consistently outperforms the two baselines in terms of optimization efficiency, reflected by its steeper fitness curves. Here we would like to clarify that we have deliberately chosen a **sufficiently large** number of robot evaluations, i.e. 1000, to hopefully allow all algorithms to converge for fair comparison. This explains why different algorithms end up with rather similar fitness levels. However, in the context of robot design automation, the convergence rate is an important aspect for evaluating design algorithms, as the evaluation of robot designs usually involves computationally expensive control learning, let alone the manufacturing costs of physical robots when deployed in real-world application. In this regard, LASeR exhibits considerable performance gains.

1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133



(a) An example of DiRect modifications on Walker-v0



(b) An example of DiRect modifications on Pusher-v0

Figure 14: Illustrative examples of DiRect modifications. Each example includes the pre- (left) and post- (right) editing designs, together with the modifications and justifications provided by an LLM to enhance morphological diversity.

Furthermore, as demonstrated in Table 4, LAsER achieves the highest diversity in two out of four tasks, while MorphVAE is dominant in the remaining two tasks. Despite the competitive performance of MorphVAE, we note that LAsER holds distinct advantages: (a) with the novel diversity reflection mechanism, LAsER is capable of achieving a **more favorable trade-off** between optimization efficiency and diversity, whereas MorphVAE proposes two variants, each of which focuses on one aspect and compromises the other; (b) MorphVAE leverages a variational autoencoder to approximate the high-performing robot distribution and generate offspring solutions, which lacks interpretability. On the contrary, LAsER can instruct an LLM to explicitly explain its design choices and thus provide valuable insights of robot design (see Appendix J). LAsER is also capable of more intelligent knowledge transfer across different tasks, utilizing the reasoning capabilities of LLMs. Furthermore, the inferior diversity outcomes of OPRO once again reveal the inefficiency of LLMs to balance exploitation with exploration on their own, and **highlight the significance of our proposed diversity reflection mechanism.**

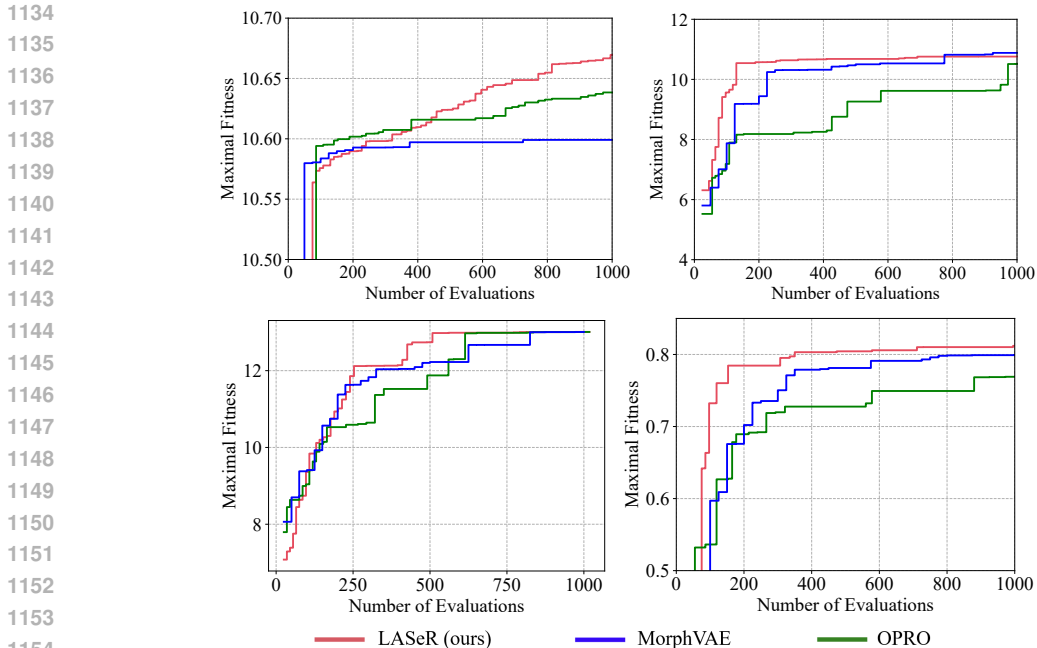


Figure 15: Comparison between LAsEr (red), MorphVAE (blue) and OPRO (green) in terms of optimization efficiency. Both MorphVAE and OPRO were implemented 3 times independently. In the original paper of MorphVAE (Song et al., 2024a), two variants are proposed that focus on either optimization efficiency or diversity. We take an average of them to reflect the overall performance of MorphVAE. This is also the case for Table 4.

Table 4: Comparison between LAsEr, MorphVAE and OPRO in terms of diversity. The results are reported as mean±standard deviation.

	Walker-v0	Carrier-v0	Pusher-v0	Catcher-v0
LAsEr (ours)	23.09 (5.33)	20.87 (4.27)	20.91 (8.85)	6.15 (1.33)
MorphVAE	16.2 (N/A)	33.16 (16.59)	18.18 (12.48)	11.00 (3.09)
OPRO	20.77 (7.34)	5.06 (2.49)	9.55 (2.92)	6.76 (N/A)

H ADDITIONAL REPEATED EXPERIMENTS

In response to reviews’ suggestions, we conducted two more sets of repeated experiments on LAsEr and LLM-Tuner (the most competitive baseline), and now present the averaged results from a total of 5 repeated experiments. As demonstrated in Figure 16, the advantageous optimization efficiency of LAsEr remains obvious. We also conduct hypothesis testing (specifically two-tailed t-test) to prove statistical significance. Since in this work we have chose a sufficiently large number of robot evaluations (i.e. 1000) to give ample opportunity to all algorithms to converge, it becomes more relevant to compare the convergence speed rather than entire fitness curves. To this end, we first average the eventual fitness values obtained by all repeated experiments (denoted as f) within a given task, and then record the number of evaluations that each experiment took to reach this average fitness (denoted as n). For those that did not reach f , n is simply recorded as 1000. We then conduct a two-tailed t -test to compare the n ’s of different algorithms. For Carrier-v0, f is 10.69, and n is on average **719.2** and **979** for LAsEr and LLM-Tuner, respectively ($p = 0.029$). For Pusher-v0, f is 12.95, and n is on average **528** and **888.6** with $p = 0.054$. For Walker-v0, since none of the experiments of LLM-Tuner reach $f = 10.65$, we instead compare the eventual fitness values achieved by LAsEr and LLM-Tuner, which are on average **10.67** and **10.63**, with $p < 0.001$. Since we are comparing against the best-performing baseline, we believe the above analysis confirms the significant advantage of LAsEr in terms of optimization efficiency.

The morphological diversity achieved by LAsER remains higher than LLM-Tuner on average (Table 5). However, we note that both LAsER and LLM-Tuner exhibit relatively high variability in their diversity outcomes, suggesting that even more repetitions would be needed to establish statistical significance. To this end, we will continue with additional repeated experiments.

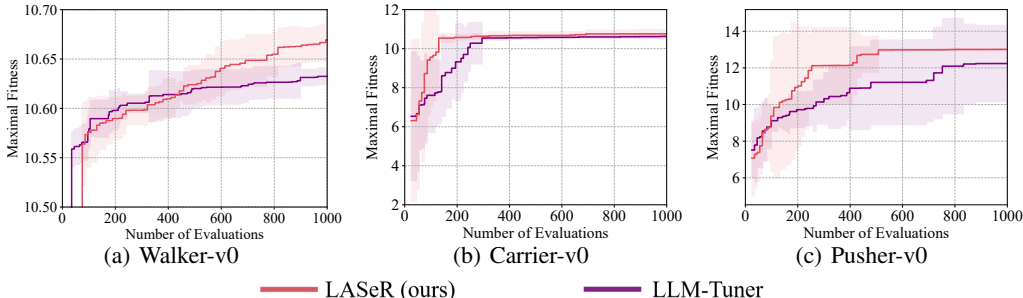


Figure 16: Comparison between LAsER (red) and LLM-Tuner (purple) based on five repeated experiments, in terms of optimization efficiency. The colored bands represent mean±1.96×standard deviation. We will continue with the remaining baselines and include the complete results in our paper once they are available.

Table 5: Comparison between LAsER and LLM-Tuner in terms of morphological diversity, based on five repeated experiments. The results are reported as mean±standard deviation.

	Walker-v0	Carrier-v0	Pusher-v0
LAsER (ours)	23.09 (5.33)	20.87 (4.27)	20.91 (8.85)
LLM-Tuner	11.60 (4.35)	18.26 (6.00)	14.17 (6.83)

I EVALUATION ON 10x10 WALKER-V0

In this work, we adhered to the standard setup used in previous VSR studies, specifically a 5x5 body size with five different materials, as this configuration is proven already expressive enough for complex and diverse morphological structures to emerge (Song et al., 2024a; Saito & Oka, 2024; Dong et al., 2023; Wang et al., 2023a;b). Nevertheless, to evaluate the scalability of our approach to larger design spaces, we tested both LAsER and LLM-Tuner (the best-performing baseline) on Walker-v0 with a 10x10 body size. Our findings, as presented in Figure 17, demonstrate that LAsER continues to outperform the baseline in terms of optimization efficiency, even in this larger design space. We attribute this success to the unique advantage of LLMs. Specifically, LLMs leverage their reasoning capabilities to identify favorable voxel assembly patterns within high-performing designs, instead of relying on random mutations (as seen in genetic algorithms and other heuristics), to generated offspring solutions. This is also demonstrated in Appendix F and J, where we show that LLMs are able to provide justifications for their decision making when generating offspring solutions and carrying out diversity reflection. It is worth noting that **the 10x10 configuration results in a design space that is 2.65×10^{52} times larger than the 5x5 case**, due to combinatorial explosion. Therefore, the promising results indicate a remarkable potential of our approach to scale to even larger and more complex robot design problems.

J LLM’S INTERPRETATION OF ROBOT DESIGN

In our preliminary experiments, we included explicit LLM reasoning, where the LLM was prompted to explain its design choices (similar to chain-of-thought). However, we observed no significant performances gains by doing so, and therefore opted to remove the reasoning process to speed up evolution and save computational costs. Nevertheless, we would like to clarify that **our approach is capable of affording higher interpretability**. This can readily be achieved by explicitly instructing the LLM to explain its decision-making process, rather than serving as a black box. We

1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295

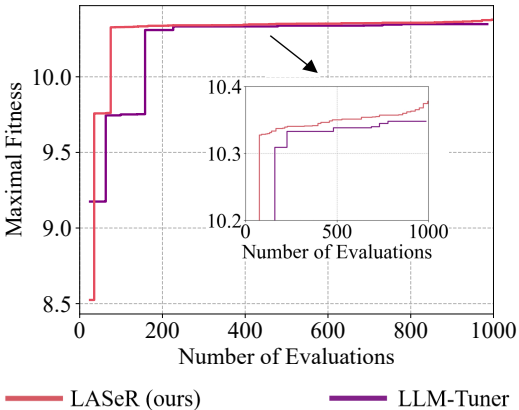


Figure 17: Comparison of LAsER (red) and LLM-Tuner (purple) on 10x10 Walker-v0. The current results are averaged across two independent runs. We plan to conduct three runs but there remains one of them unfinished. We would include the complete results once they are available. For reader’s reference, we notice that our experiments on 10x10 Walker-v0 take on average **1.5 times longer** than the 5x5 case.

provide in Figure 18 the output generated by GPT-4o-mini after receiving a collection of high-performing Catcher-v0 designs and instructed to generate a new one with higher performance. The explanations provided are insightful and reasonable, revealing the advantageous structures present in high-performing designs. This suggests that LLMs are indeed able to align specific voxel assembly patterns with high performance, and leverage these principles to generate improved offspring solutions.

K VISUALIZATION OF EVOLVED ROBOT DESIGNS

Following reviewers’ suggestions, here we provide visualizations of robot designs evolved by LAsER and baseline algorithms. We take Pusher-v0 as an example, and randomly select five high-performing robot designs obtained by each algorithm in a single run of experiment. It can be seen from Figure 19 that the robot designs evolved by LAsER seem to exhibit the highest level of variability. However, we would like to make two notes here. First, as we are only able to display a small fraction of robot designs, this visualization could only serve as a qualitative and intuitive verification of results presented in Section 4.2.1, rather than a valid means of evaluation on its own. Second, as we detailed in Appendix L, the number of evolved high-performing designs should also be taken into account when measuring morphological diversity. To this end, **we believe the quantitative results reported in Section 4.2.1 still serve as the most comprehensive and reliable evaluation of diversity.**

L FURTHER DISCUSSION ON DIVERSITY MEASUREMENT

Diversity is an important aspect for evaluating robotic systems and, in turn, the performance of robot design algorithms, as diversified design alternatives are crucial for handling dynamic environments and increasing robustness of robotic systems. To our knowledge, previous studies have predominantly employed two methods for quantifying diversity: (a) averaged measures of distinctiveness within a group of robots, such as per-voxel entropy (Song et al., 2024a) and pair-wise edit distance (Saito & Oka, 2024); (b) manual categorization of robot designs into distinct classes, followed by the calculation of the Simpson index, which is analogous to an entropy measure of class distribution (Medvet et al., 2021). The latter method becomes impractical when dealing with more abstract morphologies without clear subpopulations. The former, on the other hand, presents a **paradox** (Figure 20): including a new robot design into an existing collection can reduce diversity, even if the new design is distinct, provided that it falls within the distribution of this collection. Here, by “falling

1296 The designs provided demonstrate several key principles that contribute to their performance in the
 1297 Catcher-v0 task, where the goal is to catch and carry a box. Here's a summary of observed design
 1298 principles:
 1299 **1. Structural Stability:** The bottom row is consistently filled with rigid and soft voxels, which
 1300 provide a solid foundation and enhance stability when carrying loads. Rigid voxels are typically
 1301 positioned towards the bottom or at critical structural points to resist deformation under load.
 1302 **2. Effective Use of Actuators:** The designs incorporate both horizontal and vertical actuators
 1303 strategically. This allows for dynamic adjustments to the robot's shape to better engage with the carried
 1304 box. The placement of actuators near the center of mass or at junctions of flexible structures facilitates
 1305 effective load distribution and manipulation.
 1306 **3. Balance of Soft and Rigid Voxels:** A mix of soft and rigid voxels provides the flexibility needed
 1307 for absorbing shocks while also maintaining structural integrity. Designs often use soft voxels in upper
 1308 layers to allow for some deformation without compromising overall structure, which can help absorb
 1309 impacts when catching the box.
 1310 **4. Optimized Catching Mechanism:** Successful designs tend to have a specific configuration that
 1311 aids in maximizing the area available for catching the box. This often involves using soft voxels around
 1312 the central area to create a "cushion" effect. The presence of empty voxels strategically placed helps
 1313 optimize the weight distribution and maneuverability.
 1314 **5. Symmetry and Connectivity:** Many designs exhibit symmetrical features, which can help ensure
 1315 balanced weight distribution and stability during movement. The connectivity between adjacent voxels
 1316 supports resilience against shear forces and promotes coordinated movements, particularly important
 1317 during the catch-and-carry process.
 1318 **6. Adaptability:** The ability to dynamically adjust through the actuators enables the robot to adapt to
 1319 various scenarios—this is critical when dealing with different box sizes or weights.

1319 Here's a proposed new catcher design aimed at enhancing proficiency in the task:
 1320 proposed_design = np.array([[3, 4, 0, 2, 1],[1, 4, 3, 1, 3],[3, 2, 4, 3, 4],[4, 1, 2, 3, 1], [0, 1, 2, 1, 2]])

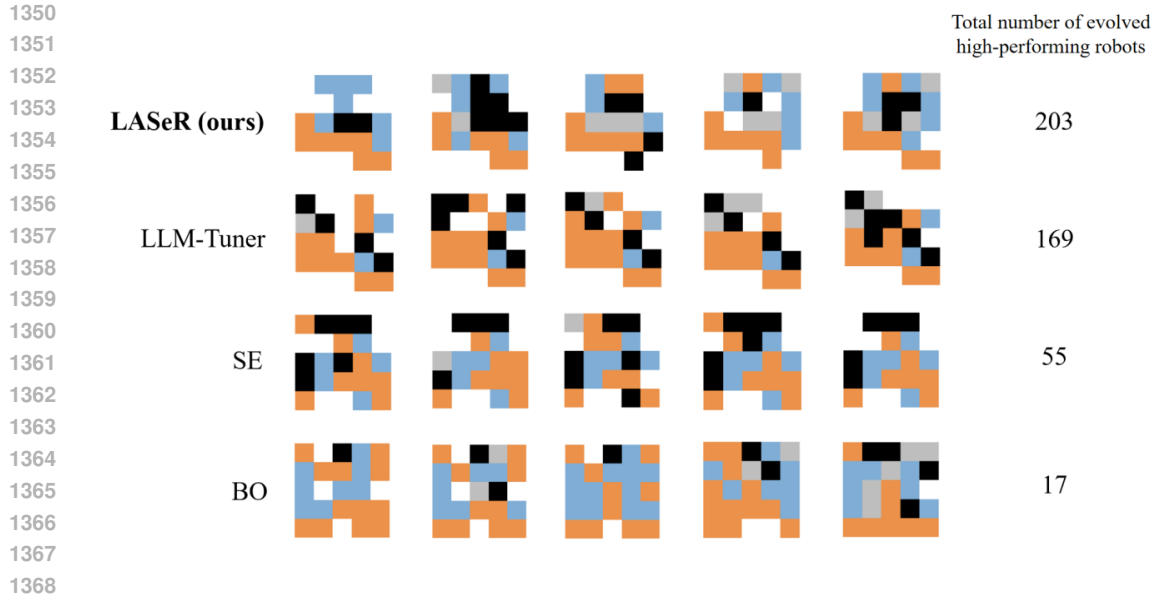
1321 **Design Features:**
 1322 **1. Enhanced Stability:** The bottom row consists of two rigid voxels flanking a soft voxel providing a
 1323 stable base while allowing slight deformation.
 1324 **2. Improved Actuation:** The second row contains two actuators (one vertical and one horizontal),
 1325 enabling more versatile manipulations and adaptations to different box sizes.
 1326 **3. Strategic Placement of Soft Voxels:** The arrangement of soft voxels in the middle provides
 1327 cushioning during the catch phase, helping to absorb any sudden movements.
 1328 **4. Central Weight Distribution:** The design emphasizes centralized soft voxels, allowing for
 1329 effective engagement and support of the box once caught.
 1330 **5. Flexibility and Adaptation:** Additional soft voxels above the actuators ensure adaptability in height
 1331 and width as needed, improving the ability to secure and carry the box efficiently.

1332 This proposed design aims to leverage the advantages of existing high-performing designs while
 1333 introducing modifications that could enhance its capacity to perform well in the 'Catcher-v0' task. By
 1334 emphasizing stability, adaptability, and an effective actuation mechanism, this design could potentially
 1335 outperform prior iterations.

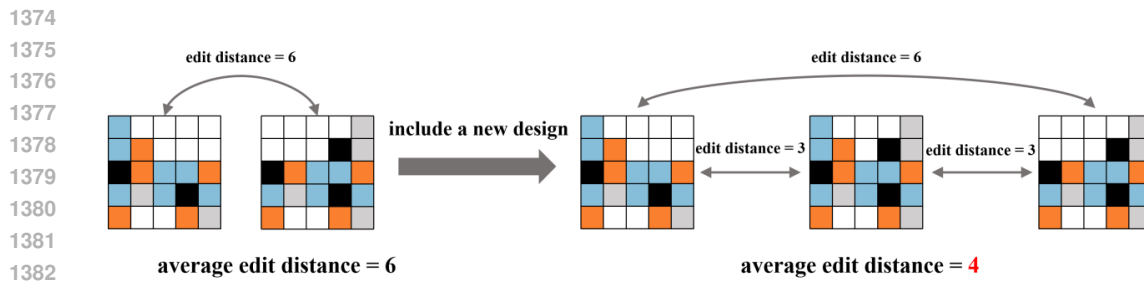
1337 Figure 18: The LLM-generated output on Catcher-v0. In this example, the LLM is prompted to
 1338 explain its decision-making process when generating a new robot design.

1340 within the distribution” we mean that the distance between the new design and existing ones is on
 1341 average smaller than that within the original collection.

1342 To address the above issue, we incorporate the number of distinct robot designs into our measure-
 1343 ment as a *correction*. Thus, our two measures – **edit distance** (measuring the distinctiveness of
 1344 evolved designs) and **the number of distinct designs** – complement each other, providing a **more**
 1345 **comprehensive and reasonable** characterization of diversity. However, we acknowledge that the
 1346 weights assigned to these quantities (1.0 and 0.1) are somewhat expedient and primarily intended
 1347 to bring them onto the same scale. This is based on our preliminary experiments where we found
 1348 that the number of distinct high-performing designs obtained in a single run of experiment typi-
 1349



1369 Figure 19: Visualizations of evolved high-performing robot designs on Pusher-v0. For each algo-
1370 rithm, five designs are randomly chosen for display. Annotated on the right are total numbers of
1371 high-performing designs obtained. The result of RoboGAN is absent because it fails to obtain robot
1372 designs that surpass the fitness threshold (i.e. the 90% quantile of fitnesses achieved by all algo-
1373 rithms).



1385 Figure 20: A paradox with diversity measurement. The inclusion of a new, distinct robot design
1386 decreases, rather than increases, the diversity when solely measured as the edit distance. This is
1387 counter-intuitive as the addition of a distinct alternative should benefit diversity.

1388
1389
1390 cally ranged from several dozens to around two hundred, while the edit distance is defined to range
1391 between 0 and 25. Given the lack of universally accepted metrics for measuring morphological
1392 diversity, we hope our approach could inspire future work to devise even more reasonable and com-
1393 prehensive approaches.

1394 We additionally provide both the separate measures of diversity (Table 6 and 7) and their weighted
1395 average (Table 8). These results also include two additional SOTA baselines and Catcher-v0 (a
1396 hard task). Our finding suggests that LASeR has more of an advantage in discovering distinct high-
1397 performers than achieving high averaged edit distance. We cannot actually state which approach
1398 is more favorable, as both benefit diversity in their own way. However, combining the results of
1399 optimization efficiency (i.e. the fitness curves), it is clear that LASeR better balances exploration
1400 with exploitation.

1401 Meanwhile, as we pointed out above, we believe that the results in Table 7 are somewhat misleading,
1402 because edit distance in itself does not suffice as a valid diversity metric. Even if a group of robot
1403 designs has another group as its subset, the former might still have a lower edit distance (even much
lower, due to the paradox in Figure 20). The comparative results in Table 7 might not be in our favor,

but we choose to display them to reveal **an open problem regarding diversity measurement** and hopefully inspire future work to investigate further.

We additionally come up with yet another diversity metric – **the total number of different voxels between all pairs of high-performing robots** (i.e. edit distance without being averaged). We believe this metric more naturally aggregates distinctiveness and number of distinct high performers without needing pre-specified weighting coefficients, thus benefiting from **better interpretability**. The results are reported in Table 9, which show that LAsER ranks first in Walker-v0 and second in the remaining tasks. The major competitor here is MorphVAE, which achieves an average rank of 2.5 across four tasks. LAsER, on the other hand, achieves 1.75. This means that, according to this newly proposed metric, LAsER still achieves the highest overall diversity.

Table 6: Comparative Results of Aggregated Diversity

	Walker-v0	Carrier-v0	Pusher-v0	Catcher-v0
BO	N/A	8.35 (N/A)	11.31 (N/A)	19.35 (1.15)
SE	5.40 (0.30)	15.84 (0.77)	6.61 (3.45)	14.13 (2.90)
RoboGAN	N/A	10.94 (N/A)	N/A	18.46 (1.74)
LLM-Tuner	11.60 (4.35)	18.26 (6.00)	14.17 (6.83)	13.89 (7.32)
MorphVAE	16.20 (N/A)	33.16 (16.59)	18.18 (12.48)	11.00 (3.09)
OPRO	20.77 (7.34)	5.06 (2.49)	9.55 (2.92)	6.76 (N/A)
LAsER(ours)	23.09 (5.33)	20.87 (4.27)	20.91 (8.85)	6.15 (1.33)

Table 7: Comparative Results of Average Edit Distance

	Walker-v0	Carrier-v0	Pusher-v0	Catcher-v0
BO	N/A	6.65 (N/A)	9.61 (N/A)	19.00 (1.00)
SE	3.95 (0.55)	12.31 (3.23)	5.87 (3.61)	13.33 (2.58)
RoboGAN	N/A	10.64 (N/A)	N/A	18.21 (1.79)
LLM-Tuner	8.23 (1.79)	12.76 (1.69)	9.54 (3.42)	13.33 (7.21)
MorphVAE	16.00 (N/A)	9.04 (3.36)	8.03 (3.72)	9.90 (4.00)
OPRO	5.52 (1.34)	4.03 (1.63)	3.92 (0.27)	5.56 (N/A)
LAsER(ours)	6.51 (0.09)	4.67 (0.34)	5.07 (0.43)	4.82 (0.67)

Table 8: Comparative Results of the Number of Distinct High-Performing Designs

	Walker-v0	Carrier-v0	Pusher-v0	Catcher-v0
BO	N/A	17.00 (N/A)	17.00 (N/A)	3.50 (1.50)
SE	14.50 (2.50)	35.33 (33.71)	7.33 (4.50)	8.00 (3.27)
RoboGAN	N/A	3.00 (N/A)	N/A	2.50 (0.50)
LLM-Tuner	33.75 (41.54)	55.00 (67.02)	46.25 (70.90)	5.67 (1.70)
MorphVAE	2.00 (N/A)	241.17 (179.83)	101.50 (133.73)	11.00 (10.18)
OPRO	152.50 (38.50)	10.33 (11.09)	56.33 (22.95)	12.00 (N/A)
LAsER(ours)	165.80 (53.09)	162.00 (43.23)	158.40 (88.68)	13.33 (8.34)

M FINER-GRAINED ABLATION ON PROMPT DESIGN

The prompt used in our study consists of three major components: task-related metadata, elite design-fitness pairs, and target fitness.

Table 9: Comparative Results of Total Edit Distance (Measured in Thousands)

	Walker-v0	Carrier-v0	Pusher-v0	Catcher-v0
BO	N/A	0.09 (N/A)	0.13 (N/A)	0.01 (0.01)
SE	0.53 (0.13)	5.08 (3.58)	0.84 (0.91)	3.92 (3.66)
RoboGAN	N/A	0.01 (N/A)	N/A	0.02 (0.01)
LLM-Tuner	1.10 (1.80)	4.09 (7.42)	2.77 (4.78)	0.02 (0.02)
MorphVAE	0.00 (N/A)	37.46 (32.06)	10.56 (18.23)	0.05 (0.07)
OPRO	7.57 (4.88)	0.06 (0.09)	0.72 (0.44)	0.04 (N/A)
LASeR(ours)	9.83 (5.45)	6.62 (3.55)	8.48 (6.24)	0.06 (0.07)

Note: LASeR ranks first in Walker-v0 and second in the remaining tasks. The major competitor here is MorphVAE, which achieves an average rank of 2.5 across four tasks. LASeR, on the other hand, achieves 1.75. This means that, according to total edit distance, LASeR still achieves the highest overall diversity.

- The **task-related metadata** primarily includes descriptions of task objectives and the simulation environment. This component is largely derived from the official documents of EvoGym (Bhatia et al., 2021), with **minimal modifications**. This metadata, which is often overlooked in previous works on LLM-aided robot design, serves two main purposes: to ground the evolutionary process in the specific context of the problem, and to facilitate the transfer of knowledge between different tasks.
- The second component consists of **elite design-fitness pairs** previously evaluated, where the designs are sorted according to their fitness in ascending order. This sorting is intended to leverage the pattern-completion capabilities of LLMs, a technique shown to be effective in prior research (Lange et al., 2024; Yang et al., 2024).
- The third component, the **target fitness** (also referred to as the “just-ask query” in Lim et al. (2024)), is introduced as a means of aligning the LLM’s outputs with our desired results.

We have demonstrated the indispensability of task-related metadata in Section 4.3.2. To further justify our prompt design and to complement the intuitive explanations provided above, we conducted finer-grained ablation studies and the results are reported in Figure 21. Specifically, we remove the following components one at a time: (a) the description of the simulation engine; (b) the description of task objectives; (c) the just-ask query (or target fitness); in this case, the LLM is simply prompted to generated robot designs with *higher* fitness; and (d) the ascending ordering of elite design-fitness pairs according to fitness. Our findings suggest that removing any of these components leads to performance drops. The just-ask query is proven the most essential, while simulation description and ordering play less important roles.

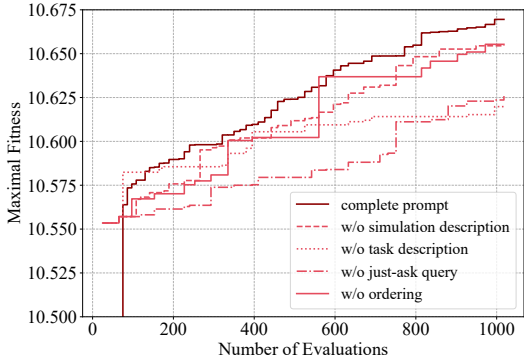


Figure 21: Finer-grained ablation studies on prompt design.

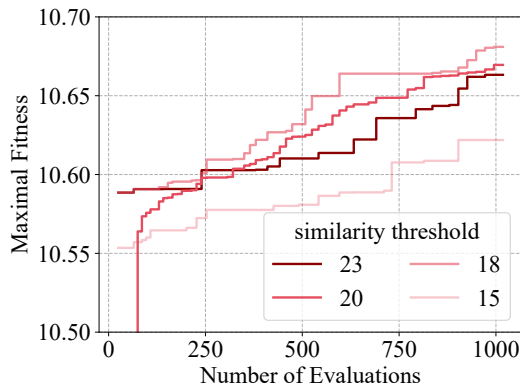
We would like to note that the phrasing of these components is **intentionally left simple and intuitive**, without applying special techniques of prompt engineering. As such, our experimental results

1512 possess a certain level of **robustness** and do not hinge on the specifics of prompt designs. However, it
 1513 would be a promising direction to integrate various prompting techniques, such as chain-of-thought
 1514 (Wei et al., 2022) and tree-of-thought (Yao et al., 2024), into our framework for better performances.
 1515

1516 N FURTHER DISCUSSION ON THE SIMILARITY THRESHOLD

1517
 1518 In our diversity reflection mechanism (DiRect), a similarity threshold is needed to decide whether a
 1519 newly generated robot design is overly similar to existing ones and therefore should undergo modi-
 1520 fications by diversity reflection. This threshold is indeed a crucial hyper-parameter that controls the
 1521 performance of LAsER. The choice of this threshold reflects the extent of diversity that one expects
 1522 to see in the evolved solutions, and hence **should be driven by the user’s specific preferences**.
 1523 For instance, setting it as 20 (as we did in our experiments) means that if a newly generated design
 1524 shares more than 20 identical voxels with any existing solution, it will be modified by DiRect to
 1525 introduce more variability.

1526 Here, we present some general principles for choosing this parameter. These principles are sup-
 1527 ported by our additional experiments with several different values of threshold (as shown in Fig-
 1528 ure 22). High similarity thresholds, like threshold=23, are generally not recommended, as they
 1529 would hinder the beneficial exploration enabled by LLMs. Conversely, excessively low thresholds
 1530 (such as threshold=15) might increase diversity but also risk overly aggressive exploration that com-
 1531 promises functionality and, in turn, harms optimization efficiency. We believe this is due to the
 1532 poor extrapolation performance of LLMs when required to propose robot designs that are much
 1533 different from given examples. Any moderate values in the middle should lead to desirable per-
 1534 formances. In fact, our findings suggest that a threshold of 18 leads to further performance gains
 1535 beyond 20, which we have chosen in our study. However, we note that lower thresholds also more
 1536 frequently trigger DiRect, which means more LLM API calls. Hence, the threshold choice also in-
 1537 volves a **trade-off between evolutionary performance** (including both optimization efficiency and
 1538 diversity) and **computational costs**, and **should be considered case by case**. We believe adaptive
 1539 threshold scheduling, based on problem specifics and evolutionary outcomes, could be a promising
 1540 direction for future research.

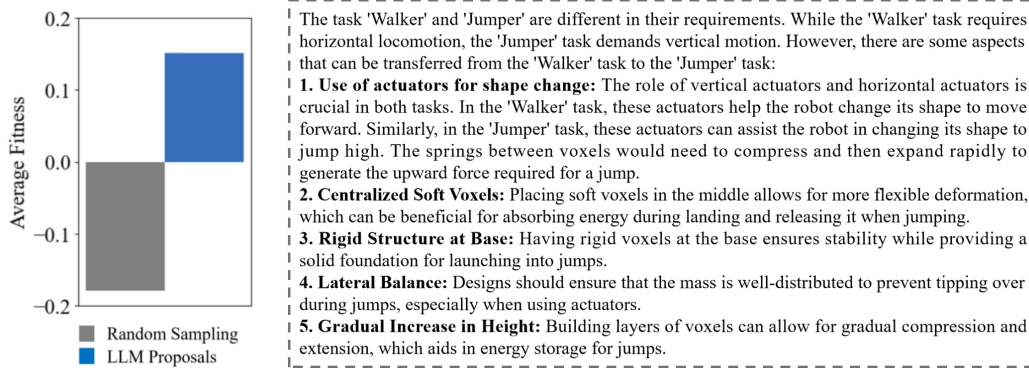


1541
 1542
 1543
 1544
 1545
 1546
 1547
 1548
 1549
 1550
 1551
 1552
 1553
 1554 Figure 22: Additional experiments with several different similarity thresholds. The diversity results
 1555 of these experiments are N/A, 28.51, 23.09, and 13.46 in the order of 15, 18, 20 and 23. The diversity
 1556 is not available for 15 because the corresponding experiment failed to generate more than one high-
 1557 performing robot designs.
 1558

1559 O GENERALIZATION ACROSS DIFFERENT TASKS

1560
 1561
 1562 In Section 4.2.2, we focused on transferring design experience between task instances that are in-
 1563 tuitively similar (i.e. Walker-v0, BridgeWalker-v0 and UpStepper-v0). These experimental designs
 1564 are largely based on the structural similarities between tasks as revealed in Wang et al. (2023b). Here
 1565 we demonstrate that this prior knowledge of task relationships is not strictly necessary for success-
 ful inter-task generalization. Specifically, we performed an additional zero-shot design experiment

(Figure 23) where the LLM was given elite samples of Walker-v0, but instructed to propose designs for a significantly different task, Jumper-v0, in which the robot is required to jump as high as possible. Despite the discrepancy between task objectives, we find that the LLM is able to dig deeper into the underlying inter-task associations and identify rather general, low-level design principles that are still relevant to the new task. Moreover, the zero-shot designs generated by LLM still outperform randomly sampled ones. See Figure 24 for visualizations of some elite Walker-v0 designs, together with a Jumper-v0 design generated by LLM. These findings suggest that LLMs possess substantial potential for generalizing experience across seemingly different optimization problems, as long as they share some common ground and are not completely irrelevant to one other (such as robot design and the Traveling Salesman Problem).



(a) Zero-shot performance (b) Design principles that LLM generalizes from Walker-v0 to Jumper-v0

Figure 23: Additional zero-shot design experiment on Jumper-v0. The zero-shot performance is averaged over 10 robot designs for both random sampling and LLM proposals. Despite the task discrepancy, the LLM is still able to identify general principles, such as the importance of actuators for shape change and rigid voxels for stability, to transfer to the new task.

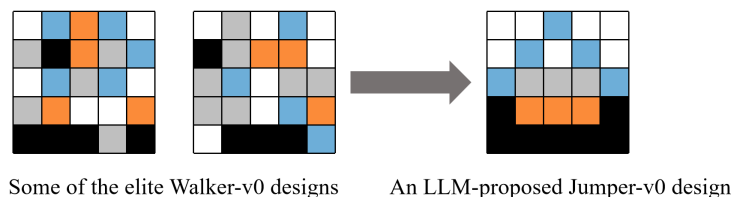


Figure 24: A visualization of some elite Walker-v0 designs (left), and the Jumper-v0 design generated by LLM (right). The Jumper-v0 design seems different than its Walker-v0 counterparts, but they still **share some underlying design principles**, such as rigid voxels in the bottom row, soft voxels in the middle rows and well-distributed mass. These features largely mirror the analysis by LLM in Figure 23.

P FURTHER DISCUSSION ON LIMITATIONS AND FUTURE WORK

In response to reviewers' suggestions, here we expand the discussion of limitations within this work, and outline several open problems that we find promising for future research.

- In this work, we developed our approach on voxel-based soft robots (VSRs), and adhered to the 5x5 body size commonly adopted in previous VSR studies. However, we would like to note that our approach is conceptually adaptable to larger design spaces (as verified in Appendix I) and potentially other types of robots. We look forward to seeing our LLM-based evolutionary framework evaluated in more extensive and complex design problems.

- 1620 • Regarding inter-task generalization, we focused on intuitively similar tasks in our paper.
1621 However, we proved that this prior knowledge of task relationships is not necessary for
1622 successful transfer of design experience (as demonstrated in Appendix O). It would be
1623 promising to investigate how this inter-task reasoning capability of LLMs could be lever-
1624 aged in a broader range of optimization problems.
- 1625 • Currently, we employed LLMs in the most cost-effective way for robot design automa-
1626 tion, that is, through in-context learning (or instruction tuning). It remains to be studied
1627 whether fine-tuning LLMs could lead to further performance gains. To our knowledge,
1628 fine-tuning Large Language Models typically involves substantially higher costs, both in
1629 terms of computational resources and the need of a sufficiently large and carefully curated
1630 dataset. However, performance might not be guaranteed due to issues like overfitting and
1631 catastrophic forgetting. Nevertheless, the prospect of a general-purpose LLM fine-tuned for
1632 various combinatorial optimization problems is intriguing, and represents an open problem
1633 for future research.
- 1634 • In this work, we followed the standard control learning approach used in previous EvoGym-
1635 based studies, i.e., PPO algorithm with MLPs serving as control networks. Since our pri-
1636 mary focus is on the evolutionary capabilities of LLMs, we chose this simple yet effective
1637 approach for fitness evaluation. That being said, a broad range of alternatives, such as
1638 value-based methods and differential simulations (Strgar et al., 2024; Cochevelou et al.,
1639 2023), could be tried out, with their impact on LLM-aided evolution examined. We also
1640 find it an interesting direction to integrate LLM-aided control strategies (such as those in
1641 Wang et al. (2023a), Brohan et al. (2023) and Cheang et al. (2024)) into our framework, so
1642 that LLMs are capable of both designing and controlling their own embodiments.
- 1643 • Morphological diversity is an important aspect for evaluating robot design algorithms, as
1644 diversified designs are crucial for ensuring the robustness of robotic systems in highly
1645 volatile environments. We pointed out the limitations of previous diversity measures in
1646 Appendix L and proposed to make a correction by taking both distinctiveness and the num-
1647 ber of distinct designs into account. We hope our work could inspire future work to develop
1648 even more reasonable and hopefully universally acceptable metrics for diversity.

1649 Q AN ANALYSIS OF COMPUTATIONAL EFFICIENCY

1650
1651 According to the data released on LLM Leaderboard (<https://artificialanalysis.ai/leaderboards/models>),
1652 for GPT-4o-mini, the median rate of output token generation is 99.8 tokens per second, and the
1653 latency (i.e. time to first token) is reported as 0.5 seconds. Given that LAsER makes an average of
1654 130 API calls per generation, with each call involving approximately 180 output tokens (here we
1655 assume the worst case where each newly generated robot design triggers DiRect), this results in an
1656 overhead of around 5 minutes per generation, or 5 hours in total.

1657 The latency issue could be mitigated with locally deployed LLMs, which are less affected by net-
1658 work delays and request queuing. However, we believe it is more pertinent to compare the overall
1659 running time of different methods, with optimization efficiency taken into account. Specifically,
1660 by checking the log messages of our programs, we find that, for Carrier-v0, in order to reach the
1661 same level of fitness, LAsER requires **7 hours**, in contrast to the most competitive baseline, LLM-
1662 Tuner, which takes **15 hours**. For Pusher-v0, the difference is greater: LAsER requires **11 hours**,
1663 whereas LLM-Tuner takes **46 hours**. On Walker-v0, LAsER is even capable of reaching a fitness
1664 **unattainable by baselines**. Hence, the rapid convergence enabled by LLMs outweighs the addi-
1665 tional computation overhead, rendering the latter perfectly worthwhile.

1666 R PSEUDO CODE OF LASER

1667
1668
1669
1670
1671
1672
1673

1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727

Algorithm 1: LAsER: LLM-Aided Evolutionary Search for Robot Design Automation

Input: A task instance T , maximum number of evaluations M , population size N , survival rate r , maximum number of LLM interactions in each generation L , probability of similarity check p , similarity threshold s .

Output: M robot designs together with their fitness scores.

Randomly initialize a population \mathcal{P} . *//Or initialize with zero-shot proposals from LLM*

$\mathcal{S} \leftarrow \mathcal{P}$ *//S keeps all the evaluated robot designs and their fitness scores*

Warm start with several generations of a traditional EA.

```

while  $|\mathcal{S}| < M$  do
   $attempts \leftarrow 0$  //Track the number of LLM interactions
   $\mathcal{P} \leftarrow$  the top  $N \times r$  robot designs among  $\mathcal{P}$ ;  $survivors \leftarrow \mathcal{P}$  //Natural selection
  while  $|\mathcal{P}| < N$  and  $attempts < L$  do
     $target\_fitness \leftarrow \max\_fitness(\mathcal{S}) \times 1.2$  //Calculate target fitness as  $1.2 \times$  current max
     $prompt \leftarrow \{\text{metadata of task } T, survivors \text{ in ascending order, } target\_fitness\}$ 
     $robot \leftarrow \text{LLM}(prompt)$  //Query the LLM for a new offspring solution
     $attempts \leftarrow attempts + 1$ 
     $u \sim \mathcal{U}(0, 1)$ 
    if  $u < p$  then
      //Enter similarity check with probability  $p$ 
      Check the similarity rate  $s'$  of  $robot$  to existing robot designs.
      if  $s' < s$  then
         $\mathcal{P} \leftarrow \mathcal{P} \cup \{robot\}$  //Pass the similarity check
      else
         $robot \leftarrow \text{DiRect}(robot)$  //Modify with Diversity Reflection Mechanism
        Check the similarity rate  $s'$  of  $robot$  to existing robot designs.
        if  $s' < s$  then
           $\mathcal{P} \leftarrow \mathcal{P} \cup \{robot\}$  //Pass the similarity check after modification
    if  $|\mathcal{P}| < N$  then
      //LLM has not generated enough robots
      Generate the remaining offspring with the traditional EA.
    for  $robot$  in  $\mathcal{P}$  do
      //Control optimization and fitness evaluation
      if  $robot$  not in  $survivors$  then
        Optimize a controller for  $robot$ . Evaluate the cumulative reward as its fitness  $f$ .
         $\mathcal{S} \leftarrow \mathcal{S} \cup \{(robot, f)\}$ 

```

Return: \mathcal{S}
