

Research Article

LLM-Supervised Adaptive Mutation Control for Evolutionary Behavior Tree Optimization in Multi-Objective Coastal Cleanup Robotics

Chi Jie Tan^{1*}, M.A. Munjer¹, Pan Wei Zheng¹, Eiji Hayashi¹, Way Soong Lim²

¹Department of Creative Informatics, Kyushu Institute of Technology, 680-4 Kawazu, Iizuka, Fukuoka, 820-0067, Japan

²Faculty of Engineering and Technology, Multimedia University, Jalan Ayer Keroh Lama, Bukit Beruang, 75450, Melaka, Malaysia

Email: tan.jie-chi339@mail.kyutech.jp, munjer.abul-md690@mail.kyutech.jp, pan.wei Zheng214@mail.kyutech.jp, haya@mse.kyutech.ac.jp, wslim@mmu.edu.my

*Corresponding Author

ARTICLE INFO

Article History

Received 1 December 2025

Accepted 20 May 2026

Keywords

Behavior tree
Large language models (LLMs)
Multi-objective optimization
Field robotics

ABSTRACT

This study investigates the use of large language models (LLMs) as adaptive supervisory controllers for evolutionary optimization of behavior trees in multi-robot systems. Unlike conventional genetic programming approaches that rely on fixed mutation and crossover parameters, the proposed framework integrates an LLM as a high-level reasoning module that dynamically adjusts evolutionary parameters based on observed population dynamics during the search process. This enables the system to maintain a balance between exploration and exploitation without requiring manual tuning of evolutionary settings. A comparative study is conducted between the proposed LLM-guided evolutionary framework and a conventional genetic algorithm with static parameters in a coastal cleanup behavior tree optimization task. The results demonstrate that incorporating LLM-based supervision leads to consistently improved optimization performance and more effective search behavior. In particular, the proposed method achieves higher-quality solutions and faster convergence to high-fitness regions compared to the baseline approach, while maintaining greater behavioral diversity throughout the evolutionary process. Further analysis shows that LLM-guided parameter adaptation enhances the generation of diverse and specialized behavioral strategies, particularly in complex multi-objective scenarios involving task execution, resource management, and environmental interaction. These findings suggest that integrating language-model-based reasoning into evolutionary search can improve both the efficiency and robustness of behavior tree optimization, offering a promising direction for adaptive robotic control system design.

© 2026 The Author(s). Published by The Society of Artificial Life and Robotics.

This is an open access article distributed under the CC BY-NC 4.0 license (<http://creativecommons.org/licenses/by-nc/4.0/>).

1. Introduction

Large Language Models (LLMs) have recently emerged as powerful computational tools capable of performing a wide range of complex tasks, particularly in natural language understanding, reasoning, and conversational interaction [1]. Although their primary success has been demonstrated in language-centric applications such as dialogue systems, summarization, and knowledge extraction, increasing attention has been directed toward their potential role as general-purpose reasoning engines within broader computational workflows [2]. Rather than being limited to text generation, LLMs are now being explored as components that can assist decision-making, coordinate processes, or provide adaptive guidance to algorithmic systems.

In parallel, Genetic Programming (GP) [3] remains an effective evolutionary computation technique for

automatically synthesizing interpretable program structures, including symbolic expressions, controllers, and behavior trees. Behavior trees (BTs), in particular, have gained popularity in robotics due to their modularity, transparency, and suitability for representing complex task-oriented behaviors. However, conventional GP approaches often encounter several practical challenges, including premature convergence, loss of population diversity, and strong sensitivity to parameter settings such as mutation and crossover rates. These issues become increasingly problematic in multi-robot environments, where the search space expands rapidly and the evolved behaviors must satisfy multiple functional requirements while maintaining coordination among agents.

Recent studies have begun exploring the integration of large language models into optimization and search processes. In several approaches, LLMs are used to generate candidate solutions, evaluate intermediate outputs, or provide heuristic guidance during iterative optimization

procedures. These hybrid frameworks demonstrate that language models can assist traditional search algorithms by proposing new solution candidates or refining existing ones based on feedback from the optimization process [4].

However, directly generating structured robotic control policies such as behavior trees using LLMs remains challenging. Multi-robot behavior trees inherently involve multiple interacting and often competing objectives. In the coastal cleanup scenario considered in this study, these objectives include: (1) efficient garbage collection from the environment, (2) collection of garbage from humanoid agents acting as secondary carriers, (3) timely delivery of collected waste to designated gathering points, (4) effective energy management through battery recharging, and (5) environmental patrolling to ensure continuous task coverage. These objectives are interdependent and may conflict with one another; for example, prioritizing aggressive garbage collection may lead to increased energy consumption, while excessive focus on recharging may reduce task productivity.

Consequently, the optimization process must balance these objectives to achieve robust and sustainable multi-robot behavior. This multi-objective nature significantly increases the complexity of the search space, making it challenging for conventional evolutionary methods and purely generative approaches to consistently produce well-balanced and high-performing behavior trees. Recent work on LLM-based optimization also indicates that handling complex multi-objective search spaces remains difficult for LLM-driven approaches without additional structured search mechanisms [5]. As a result, purely generative approaches may struggle to consistently produce valid and high-performing behavior trees, particularly when the solution space involves numerous constraints and combinatorial interactions.

An alternative strategy is to leverage the reasoning capabilities of LLMs not for direct policy generation, but for guiding the evolutionary search process itself. By observing population-level dynamics and interpreting optimization trends, LLMs can provide adaptive interventions that help regulate evolutionary parameters and maintain a healthy balance between exploration and exploitation. Despite this potential, the systematic use of LLMs as supervisory components within evolutionary algorithms has received relatively limited attention.

Motivated by this opportunity, this study proposes an LLM-guided evolutionary framework for the optimization of behavior trees in multi-robot systems. In the proposed approach, the LLM functions as a purely supervisory module that monitors evolutionary signals and dynamically adjusts parameters such as mutation intensity and diversity management strategies. By combining the global search capabilities of genetic programming with the contextual reasoning ability of LLMs, the framework aims to improve convergence behavior, preserve behavioral diversity, and ultimately produce higher-quality robotic control policies. The remainder of this paper describes the architecture of the proposed supervisory mechanism, the

adaptive strategies used to influence evolutionary dynamics, and experimental evaluations demonstrating the effectiveness of the approach in complex multi-robot task scenarios.

2. Methodology

2.1. Unity Simulation for Beach Cleaning Activities

The experimental environment was implemented using Unity (Version 6000.0.32f1) [6]. The simulation represents a large-scale coastal cleanup scenario designed to evaluate the evolution of robot behavior trees in a controlled yet realistic setting. As illustrated in Figure 1, the virtual environment models a rectangular beach area measuring approximately $2 \text{ km} \times 1 \text{ km}$. One side of the environment is bounded by an open sea region, providing a natural boundary that emulates shoreline constraints while maintaining a structured testing space for robot operation.

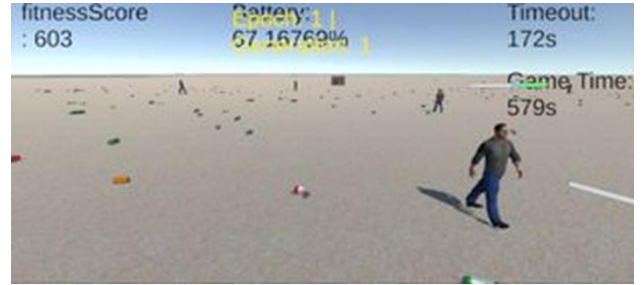


Figure 1 Beach Environment Simulation

2.1.1. Humanoid Agents

At the beginning of each simulation episode, eight autonomous humanoid agents are randomly distributed across the beach. These agents represent human participants engaged in coastal cleanup activities. Each humanoid follows a simple behavioral model consisting of random exploration, short-range perception, and basic garbage handling actions. When a piece of garbage enters the humanoid's perception range (3 m radius), the agent approaches the object and performs a pickup action. After collecting garbage, the humanoid transfers it either to the cleaning robot or directly to the clearing station depending on the relative proximity of these two entities at the time of decision-making.

2.1.2. Cleaning Robot

A single autonomous cleaning robot is initialized near the center of the environment. The robot operates using a behavior tree that controls multiple functional modules, including area patrolling, direct garbage pickup from the ground, garbage collection from humanoid agents, disposal at the clearing station, and battery management through docking at a charging station. The robot is capable of detecting both garbage items and humanoid agents within a sensing radius of 10 m, enabling responsive interaction with its surroundings.

2.1.3. Environmental Objects

To support the cleanup task, a combined clearing and charging station is located at one end of the beach and serves as the operational hub for garbage disposal and battery recharging. In addition, 200 garbage items are randomly distributed throughout the environment at the start of each simulation episode. These items represent common forms of marine litter such as broken bottles, glass fragments, and other debris typically found in coastal environments.

2.2. Behavior Tree Architecture

Behavior Trees (BTs) are widely used in autonomous systems as a structured framework for representing complex decision-making processes. Compared with traditional control architectures such as finite state machines, BTs provide improved modularity, scalability, and interpretability, making them particularly suitable for robotic applications that require flexible and reactive behavior control [7]. Their hierarchical structure allows complex tasks to be decomposed into smaller reusable components while maintaining a clear execution flow.

In this study, the behaviors of both the cleaning robot and humanoid agents are implemented using the NPBehave library (Version 1.3.0) [8], which provides a lightweight implementation of BTs within the Unity simulation environment. The constructed behavior trees utilize several standard node categories, including control-flow nodes, decorators, and action nodes. Control-flow nodes such as Selector, Sequence, and Parallel govern task ordering and decision branching. Decorator nodes—including Condition, Inverter, Repeater, and Succeeder—modify execution logic or enforce behavioral constraints. Action nodes represent executable robot capabilities such as patrolling, approaching targets, picking up garbage, transferring collected items, recharging at the docking station, and clearing garbage at the disposal point. Through the combination of these node types, the BT framework enables structured task prioritization and responsive interaction with environmental events.

To enable evolutionary optimization of robot behaviors, a subset of these BT components is exposed within the genetic programming (GP) framework. The evolutionary process operates on tree-structured representations composed of the defined node set while maintaining strict structural constraints. A typed-tree representation is employed to ensure syntactic correctness during genetic operations. Under this scheme, control-flow nodes may only contain child subtrees, decorator nodes operate on a single child node, and action nodes are restricted to leaf positions within the tree. These typing rules prevent the generation of structurally invalid behavior trees during crossover and mutation while preserving a sufficiently large search space for discovering effective control strategies.

By combining a structured BT representation with constrained evolutionary operators, the proposed

framework supports the automatic synthesis of diverse and interpretable robot behaviors suitable for complex multi-agent cleanup tasks. Figure 2 shows an example structure of an evolved behavior tree controlling the cleaning robot.

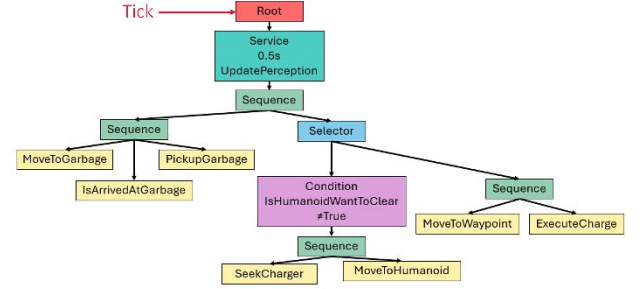


Figure 2 Example Structure of an Evolved Behavior Tree

2.3. Multi-Objective Problem Formulation

In the proposed framework, the behavior tree optimization process is formulated as a multi-objective problem, where each candidate solution must simultaneously satisfy several functional requirements within the coastal cleanup scenario. Specifically, the objectives are defined as follows:

- Garbage Collection from Environment, O_1 : Maximize the amount of garbage collected directly from the ground.
- Humanoid-Assisted Collection, O_2 : Maximize the amount of garbage retrieved from humanoid agents.
- Garbage Delivery Efficiency, O_3 : Maximize the successful delivery of collected garbage to designated gathering points.
- Energy Sustainability, O_4 : Minimize energy depletion by encouraging timely recharging behavior.
- Environmental Coverage, O_5 : Maximize patrolling effectiveness to ensure continuous exploration and task coverage.

These objectives can be grouped into three categories: (i) task execution (O_1, O_2, O_3), (ii) resource management (O_4), and (iii) environmental exploration (O_5). Importantly, these objectives are interdependent and may exhibit trade-offs. For example, aggressive garbage collection may increase energy consumption, while excessive focus on recharging may reduce overall task efficiency. To evaluate candidate behavior trees, the objectives are aggregated into a composite fitness function shown in Eq. (1):

$$F = \sum_{i=1}^5 w_i O_i \quad (1)$$

Where w_i represents the weight assigned to each objective. The weights w_i are empirically determined to balance task execution, energy efficiency, and exploration

performance in the coastal environment. In this study, the weights are empirically set as $w_1 = 1.8$, $w_2 = 1.0$, $w_3 = 1.8$, $w_4 = 1.6$, and $w_5 = 1.2$. These values reflect the relative importance of task execution and sustainability within the coastal cleanup mission, with higher priority assigned to direct garbage collection and delivery performance.

2.4. LLM-Supervised Genetic Programming

This study proposes an evolutionary optimization framework in which a large language model (LLM) supervises the evolution of robot behavior trees. Conventional genetic programming approaches typically rely on static parameter settings, such as fixed mutation rates and predefined selection policies. While simple to implement, these fixed configurations often lead to premature convergence, loss of behavioral diversity, and limited exploration of the solution space. These issues are particularly problematic in multi-robot systems, where the evolutionary search must simultaneously satisfy multiple behavioral objectives.

To address these limitations, the proposed framework incorporates an LLM as a supervisory component that dynamically adjusts evolutionary dynamics during the optimization process. The system employs Claude Haiku 4.5 [9] as a meta-controller that continuously evaluates the state of the evolving population and provides adaptive guidance. LLM influences the evolutionary process through an adaptive parameter control, where mutation rates are adjusted based on observed population dynamics. With this mechanism, the LLM helps maintain a balance between exploration and exploitation during the evolutionary search. The overall architecture forms a closed-loop system consisting of four main components: a population evaluator, a pattern detector, the LLM meta-controller, and the evolutionary engine, as illustrated in Figure 3.

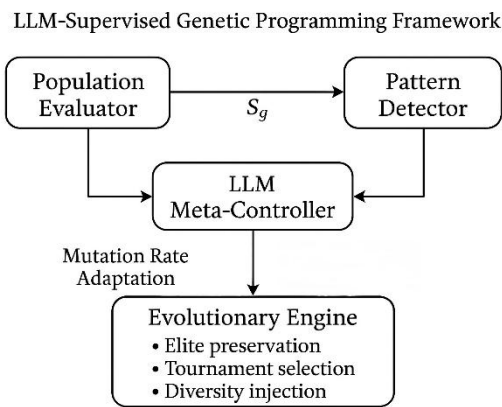


Figure 3 LLM-Supervised GP System Overview

2.4.1. Population State Representation

At each generation g , the population evaluator summarizes the evolutionary state using a structured state vector, S_g . This vector captures multiple indicators describing both performance and diversity within the

population. Fitness-related statistics include the maximum fitness, mean fitness, and standard deviation of the current generation. In addition, quality indicators record the number of individuals classified as excellent or complete, based on predefined behavioral performance thresholds.

Behavioral diversity is quantified using the Herfindahl–Hirschman Index (HHI) [10], computed over the distribution of behavioral specialists present in the population. Five behavioral niches are considered in the system: pickup behavior, clearing behavior, charging behavior, humanoid interaction behavior, and balanced multi-behavior strategies. Let D denote the number of direct garbage pickup actions and H denote the number of humanoid interaction actions recorded for an individual over a fixed evaluation horizon. The classification rules are defined as follows:

- **Pickup Specialist:** if $D > 2H$, indicating a strong preference toward direct environmental garbage collection.
- **Humanoid Interaction Specialist:** if $H > 2D$, indicating dominant interaction with humanoid agents.
- **Balanced Behavior:** otherwise, representing individuals that do not exhibit a strong dominance toward either pickup or humanoid interaction behavior.

In addition, two functional capability-based classifications are defined based on the occurrence of specific actions during evaluation:

- **Charging Specialist:** assigned if at least one battery charging action is executed.
- **Humanoid Interaction Specialist:** assigned if at least one delivery action to a designated clearing or gathering point is executed.

An individual may belong to multiple functional categories (e.g., Charging and Pickup), while dominance-based categories (Pickup, Humanoid, Balanced) are mutually exclusive. This classification scheme is used to construct the behavioral distribution, B_g for diversity measurement via the Herfindahl–Hirschman Index. The diversity score D_g is defined using Eq. (2):

$$D_g = 1 - \text{HHI}(B_g) \quad (2)$$

Where B_g represents the distribution of specialist types in generation g . Additional information, including specialist counts and a five-generation historical trajectory of population metrics, is incorporated to provide temporal context for trend analysis. Eq. (3) shows the HHI formulation:

$$\text{HHI}(B_g) = \sum_{i=1}^5 p_i^2 \quad (3)$$

Where p_i represents the proportion of individuals belonging to the i -th behavioral niche in generation g ,

corresponding to the five categories: pickup, clearing, charging, humanoid interaction, and balanced behavior. A higher value of D_g indicates greater behavioral diversity, while a lower value indicates dominance of a small number of behavioral niches.

2.4.2. Evolutionary Pattern Detection

Before interacting with the LLM, a lightweight rule-based classifier analyzes the population statistics to identify the current evolutionary regime. This pattern detector categorizes each generation into several states, including HealthyProgress, DiversityCrisis, ExplorationPhase, RapidImprovement, or RecoveryPhase. These regimes are determined based on conditions such as stagnation in fitness growth, collapse of behavioral diversity, or the emergence of high-quality solutions. This intermediate classification step provides contextual guidance for the LLM by filtering noisy early-generation signals and highlighting meaningful evolutionary trends.

2.4.3. LLM Meta-Controller

The LLM meta-controller receives both the population state vector and the detected evolutionary regime as input. Based on this information, it produces supervisory actions that regulate the evolutionary process. The primary output of the LLM is an adaptive mutation rate, generated at each generation to reflect the current optimization conditions.

The prompt design encourages the LLM to reason about population stability and long-term optimization performance. During early stages of evolution, populations may exhibit artificially high fitness values due to stochastic effects rather than genuine behavioral competence. To mitigate this issue, the LLM is instructed to maintain relatively high mutation rates until stable high-quality individuals emerge. As the evolutionary process progresses and reliable behaviors begin to appear, the mutation rate is gradually reduced to support refinement of promising solutions. The LLM outputs a mutation rate within the interval $[0.1, 0.5]$, accompanied by a short explanation and a confidence assessment. This structured response provides transparency into the reasoning process behind each supervisory decision.

More generally, the LLM-guided parameter control operationalizes the exploration–exploitation trade-off through real-time population diagnostics. When indicators of evolutionary stagnation are detected—such as fitness improvement below 1% over five generations, diversity index values below 0.3, or the disappearance of key behavioral specialists—the system increases mutation intensity to reintroduce genetic variation and escape local optima. Conversely, when the population demonstrates stable progress, balanced behavioral niches, and diversity values above 0.5, the mutation rate is reduced to facilitate fine-grained optimization within productive regions of the search space.

2.4.4. Evolutionary Engine and Diversity Protection

The evolutionary engine integrates the LLM-generated supervisory signals into a modified genetic programming pipeline. Each generation maintains a population size of 30 individuals and follows a hybrid reproduction strategy consisting of elite preservation, tournament-based reproduction, and diversity injections.

The best-performing individuals are preserved to maintain continuity of high-quality solutions, while most offspring are produced through tournament selection. To ensure continued exploration, a fixed portion of each generation is reserved for randomly generated individuals, which serve as diversity injectors. The mutation intensity applied to individuals varies according to their origin: elite individuals undergo reduced mutation, tournament offspring receive standard mutation rates, and randomly generated individuals are subjected to higher mutation levels. The exact details are summarized in Table 1.

Table 1 Population Structure and Mutation Rate Multiplier

Individual Type	Index Position	Mutation Rate	Rationale
Elite	0-2	0.3x	Preserve strong solutions
Tournament -selected	3-23	1.0x	Standard variation
Random diversity	24-29	2.0x	Aggressive exploration

To further prevent evolutionary collapse, the framework incorporates a niche preservation mechanism. First, the best individual from each critical behavioral niche—charging, clearing, and humanoid interaction—is automatically retained each generation. Second, if a behavioral niche disappears entirely, the most recent representative stored in the historical archive is reintroduced into the population. These strategies maintain behavioral diversity and ensure that important capabilities are not permanently lost during the evolutionary process. The whole process is formulated as in Algorithm 1.

Algorithm 1: LLM-Supervised Genetic Programming

```

1 Initialize population  $P_0$ 
2 Set mutation rate  $\mu \leftarrow \mu_0$ 
3 for generation  $g = 1$  to  $G$  do
4   Evaluate fitness and behavioral metrics of all
     individuals in population  $P_{t-1}$ 
5   Compute population state vector  $S_g$ 
6   Detect evolutionary regime  $R_g$ 
7   Query LLM  $(S_g, R_g)$  for mutation rate  $\mu_g$ 
8   Generate offspring using GP operators with
     suggested mutation rate,  $\mu_g$ 
9   Apply niche protection and diversity injection
10  Form next population  $P_{g+1}$ 
11 end for
12 Return evolved population  $P_g$ 

```

3. Experimental Result

3.1. Comparative Performance and Convergence Analysis

To assess the effectiveness of the proposed LLM-guided mutation control strategy, the evolutionary performance of the framework was compared with a baseline genetic programming configuration using a fixed mutation rate. Both approaches were evaluated under identical experimental conditions, including the same simulation environment, population size ($N = 30$), and evolutionary duration of 48 generations. This controlled setup ensures that performance differences arise primarily from the adaptive supervisory mechanism rather than external experimental factors.

Table 2 the primary performance metrics obtained from both systems. The LLM-supervised framework consistently achieved higher fitness levels throughout the evolutionary process. In particular, the maximum fitness achieved during evolution increased by 16.3%, while the average of the best individuals across generations improved by 17.4%. In addition, the average fitness of the final generation was 14.7% higher than that obtained using the baseline configuration. These improvements indicate that the adaptive mutation strategy guided by the LLM not only enhances peak solution quality but also improves the overall evolutionary trajectory.

Table 2 Aggregate Performance Comparison

Metric	LLM-Supervised	Baseline	Improvement
Peak Fitness	47,689	41,011	+16.3%
Average Max Fitness	35,947	30,627	+17.4%
Final Max Fitness	44,508	40,849	+9.0%
Final Average Fitness	19,409	16,927	+14.7%
Excellent Robots	423	167	+153.3%
Complete Robots	23	148	-84.5%

Further insight can be obtained from the convergence characteristics summarized in Table 3. Although the baseline configuration reached the intermediate fitness threshold of 30,000 slightly earlier (generation 11) than the LLM-supervised system (generation 13), the adaptive approach subsequently demonstrated significantly faster progress toward higher performance levels. After this point, the LLM-guided system rapidly surpassed the baseline and continued improving until achieving fitness values exceeding 45,000 by generation 19. In contrast, the baseline configuration never reached this performance level within the allotted evolutionary period. Overall, these results suggest that the LLM-supervised framework provides a more effective balance between exploration and exploitation during the evolutionary search. While early

progress may appear slower due to sustained diversity, the adaptive mutation policy ultimately enables more stable convergence and the discovery of higher-quality behavior tree solutions for the multi-robot task.

Table 3 Convergence Threshold Comparison

Fitness Threshold	LLM-Supervised	Baseline	Speedup
30,000	Gen 13	Gen 11	-18.2%
35,000	Gen 13	Gen 24	+45.8%
40,000	Gen 14	Gen 32	+56.3%
45,000+	Gen 19	Not achieved	—

3.2. Early Evolution Dynamics

To better understand how the LLM-supervised framework influences the evolutionary search process, we examined the progression of population fitness during the early generations of evolution. Table 4 summarizes the best fitness values observed at selected generations during the first 20 generations for both the LLM-supervised and baseline configurations. The results reveal a distinctive convergence pattern between the two approaches. During the initial phase of evolution (generations 1–12), the baseline configuration demonstrates slightly faster fitness improvements. For example, by generation 10 the baseline population reached a fitness value of 29,195, while the LLM-supervised system achieved 28,001. This difference suggests that the fixed-mutation baseline tends to exploit early high-performing individuals more aggressively.

Table 4 Early-Generation Fitness Progression

Generation	LLM-Supervised	Baseline	Fitness Difference
1	25,313	25,313	0
5	25,323	25,313	+10
10	28,001	29,195	-1,194
13	37,734	30,975	+6,759
14	40,951	31,070	+9,881
15	34,417	25,382	+9,035
19	44,483	32,041	+12,442
20	37,736	29,266	+8,470

However, the evolutionary trajectories begin to diverge substantially after generation 12. Starting from generation 13, the LLM-supervised system exhibits a rapid acceleration in fitness improvement, surpassing the baseline by a significant margin. At generation 13, the best individual in the LLM-supervised population achieved a fitness of 37,734, compared with 30,975 in the baseline population. This gap continued to widen in subsequent generations, reaching a difference of +12,442 by generation 19.

This delayed but sharper improvement indicates that the adaptive mutation policy maintained a higher level of genetic diversity during the early stages of evolution. By sustaining exploration rather than immediately converging toward early solutions, the LLM-supervised system was able to discover more promising regions of the search

space. Once high-quality behavioral structures emerged, the mutation rate was gradually reduced, allowing the evolutionary process to shift toward exploitation and refinement of these promising solutions. Overall, the observed convergence dynamics suggest that the LLM-guided mutation strategy effectively regulates the exploration–exploitation balance throughout the evolutionary process. The initial exploration phase helps prevent premature convergence, while the later exploitation phase enables rapid performance improvement once viable behavioral patterns have been identified.

3.3. Behavioral Diversity and Specialist Stability

Figure 4 illustrates the differences in population composition between the LLM-supervised evolutionary framework and the fixed-mutation baseline. Figure 4A shows the distribution of primary behavioral specialists, which are mutually exclusive categories representing the dominant behavior exhibited by each evolved robot. Figure 4B summarizes secondary behaviors and quality indicators, which may coexist with any primary specialization.

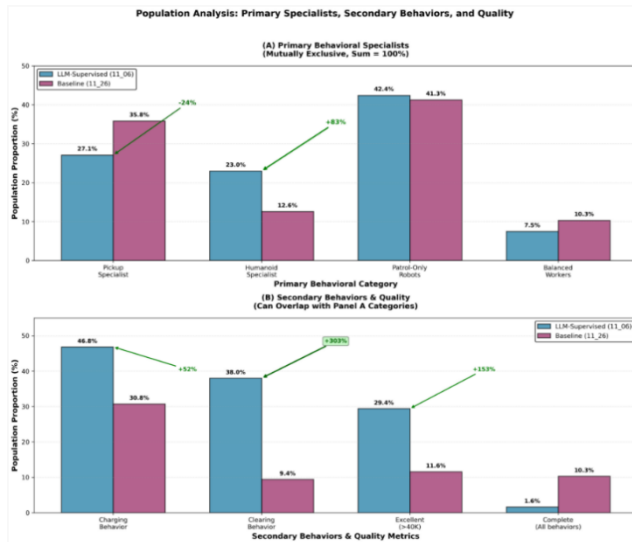


Figure 4 Population Analysis

As shown in Figure 4A, the LLM-supervised population exhibits a more balanced distribution of behavioral roles. In the baseline condition, pickup specialists dominate the population, accounting for 35.8% of individuals. Under LLM supervision, this proportion decreases to 27.1%, representing a 24% reduction in pickup dominance and mitigating the tendency toward a single-behavior monoculture. At the same time, the proportion of humanoid interaction specialists increases from 12.6% to 23.0%, corresponding to an 83% relative increase. This change indicates that the adaptive mutation strategy successfully preserves more complex behavioral patterns that would otherwise disappear under strong selection pressure. Patrol-only robots remain relatively stable across both configurations (42.4% vs. 41.3%), while balanced

multi-role robots appear at low but comparable frequencies (7.5% vs. 10.3%). Overall, these results suggest that the LLM-guided evolutionary process maintains a richer distribution of behavioral strategies across the population.

Additional improvements are observed in the emergence of functional task behaviors, as shown in Figure 4B. Charging behavior appears in 46.8% of individuals under LLM supervision, compared with 30.8% in the baseline population, representing a 52% increase. Even more pronounced improvements are observed for clearing behavior, which rises from 9.4% in the baseline population to 38.0% under LLM supervision, corresponding to a 303% increase. These results indicate that the adaptive mutation policy significantly improves the discovery and retention of behaviors necessary for effective multi-task operation.

Quality-oriented metrics further highlight the advantages of the proposed approach. The proportion of excellent robots, defined as individuals achieving fitness values above 40,000, increases from 11.6% in the baseline system to 29.4% in the LLM-supervised population, representing a 153% improvement in elite solution production. In contrast, the proportion of robots classified as complete-behavior individuals is lower in the LLM-supervised system (1.6% vs. 10.3%). This difference suggests that the baseline tends to converge toward solutions that satisfy minimal behavioral requirements, whereas the LLM-guided framework continues to push the population toward higher-performance implementations of these behaviors.

Taken together, these results demonstrate that LLM-guided mutation control not only prevents the collapse of behavioral diversity but also promotes the emergence of more capable and specialized robotic behaviors. By maintaining a balanced set of behavioral niches while encouraging higher-quality solutions, the proposed framework enables more robust multi-objective optimization in complex robotic environments.

4. Conclusion

This study investigated the integration of large language model supervision into a genetic programming framework for evolving behavior trees in a multi-robot coastal cleanup scenario. By allowing the LLM to dynamically regulate mutation rates and support diversity preservation during evolution, the proposed framework introduces an adaptive supervisory mechanism that addresses several limitations of conventional fixed-parameter evolutionary algorithms.

Experimental results demonstrate that the LLM-guided approach consistently outperforms the baseline configuration across multiple evaluation metrics. Over 48 generations of evolution, the proposed method achieved improvements of approximately 16–17% in peak and average maximum fitness, while reaching high-performance thresholds 45–56% faster than the fixed-mutation baseline. In addition, the LLM-supervised system generated 153% more high-quality (“excellent”) solutions,

indicating a broader and more effective exploration of the solution space.

Analysis of population composition further revealed that the adaptive mutation strategy helped maintain a richer set of behavioral specializations throughout the evolutionary process. In particular, complex roles such as humanoid interaction specialists were retained at significantly higher levels, and task-critical behaviors such as clearing operations increased substantially. These findings suggest that the supervisory mechanism effectively mitigates premature convergence and supports the emergence of diverse and functionally capable robotic behaviors.

Overall, the results demonstrate that reasoning-driven supervisory signals from large language models can effectively complement traditional fitness-based optimization, enabling more adaptive evolutionary dynamics in complex search spaces. This work highlights the potential of LLMs as high-level supervisory controllers for evolutionary behavior tree optimization. Future work will investigate direct injection of LLM-generated behavior trees into the evolutionary population to further enhance behavioral diversity and search efficiency.

Acknowledgements

Not Applicable.

References

1. OpenAI, "GPT-4 technical report," *arXiv*, 2024.
2. Chengrun Yang and Xuezhi Wang and Yifeng Lu and Hanxiao Liu and Quoc V. Le and Denny Zhou and Xinyun Chen, "Large Language Models as Optimizers," *arXiv*.
3. Koza, John R., Genetic programming: on the programming of computers by means of natural selection, Cambridge: MIT Press, 1992.
4. Sen Huang, Kaixiang Yang, Sheng Qi, and Rui Wang, "When large language model meets optimization," *Swarm and Evolutionary Computation*, vol. 90, 2024.
5. Andrej Schwanke, Lyubomir Ivanov, David Salinas, Frank Hutter, Arber Zela, "Multi-Objective Hierarchical Optimization with Large Language Models," *ArXiv Machine Learning*, 2026.
6. U. Technologies, "Unity," 2023. [Online]. Available: <https://unity.com/>. [Accessed 2025].
7. Michele Colledanchise, Petter Ögren, "Behavior Trees in Robotics and AI," 2017.
8. Meniku, "NPBehave - An event driven Behavior Tree Library for code based AIs in Unity," Github, 2016. [Online]. Available: <https://github.com/meniku/NPBehave>. [Accessed 2025].
9. Anthropic, "Claude Haiku 4.5," 2025. [Online]. Available: <https://www.anthropic.com/claude/haiku>. [Accessed 2025].
10. P. D. Waggoner, "The hhi package: Streamlined calculation and visualization of Herfindahl-Hirschman Index scores," *Journal of Open Source Software*, vol. 3, no. 28, p. 828, 2018.

Authors Introduction

Mr. Chi Jie Tan



He received his Bachelor of Engineering Electronics Majoring in Robotics and Automation from the Faculty of Engineering, Multimedia University, Malaysia in 2020. He received Master of Creative Informatics and is currently pursuing for Ph.D. degree at Kyushu Institute of Technology.

Mr. M.A. Munjer



He received his Masters of Science in Engineering from the Department of Electrical and Electronic Engineering, RUET, Bangladesh in 2019. He is currently a Doctoral student at Kyushu Institute of Technology.

Mr. Pan Wei Zheng



He received his Bachelor's degree in Software Engineering from the Department of Computer Science, Qingdao University, China in 2022. He is currently a Master's student at Kyushu Institute of Technology at Hayashi Laboratory.

Prof. Eiji Hayashi



Prof. Eiji Hayashi is a professor in the Department of Intelligent and Control Systems at Kyushu Institute of Technology. He received the Ph.D. (Dr. Eng.) degree from Waseda University in 1996. His research interests include Intelligent mechanics, Mechanical systems and Perceptual information processing. He is a member of The Institute of Electrical and Electronics Engineers (IEEE).

Dr. Lim Way Soong



He is a Professor at the Faculty of Engineering and Technology, Multimedia University, Malaysia. He received his Ph.D. degree (Engineering) from Multimedia University of Malaysia, in 2007. His research interest covers areas in artificial intelligence, neural network, pattern recognition, road traffic monitoring and robotics navigation.
