DOI: 10.21917/ijsc.2025.0546

A HYBRID BIO-INSPIRED ALGORITHM FOR ROBUST GLOBAL OPTIMIZATION USING GAZELLE-DIFFERENTIAL EVOLUTION IN COMPLEX ENGINEERING DOMAINS

V. Sabaresan[1] and P. Kumari[2]
[1]*Department of Computer Science and Engineering, St. Joseph's Institute of Technology, India*
[2]*Department of Computer Science and Engineering, Excel Engineering College, India*

Abstract

In complex engineering systems, solving high-dimensional, nonlinear, and multimodal optimization problems remains a formidable challenge. Traditional optimization techniques often converge prematurely or fail to scale effectively with problem complexity. Nature-inspired metaheuristics, such as Differential Evolution (DE) and Gazelle Optimization Algorithm (GOA), have shown promise in addressing such issues due to their adaptive exploration and exploitation capabilities. While DE excels in global exploration through mutation and crossover strategies, it suffers from limited convergence precision in rugged landscapes. Conversely, the Gazelle Optimization Algorithm, inspired by the evasive and coordinated movement of gazelles under predation, provides better adaptability in exploitation but lacks the stochastic diversity for broad search spaces. Thus, combining the strengths of both may overcome their individual limitations. This paper proposes a novel hybrid approach termed Gazelle-Differential Evolution (GoDE). The algorithm synergistically integrates the exploitation ability of GOA with the exploration strength of DE. Specifically, GoDE leverages gazelle dynamics for local refinement and DE's differential mutation for global search. A dynamic control parameter regulates the hybridization intensity, ensuring a balanced optimization process. GoDE was evaluated on 25 benchmark functions (CEC 2023) and three real-world engineering design problems (pressure vessel, welded beam, and hydro-turbine blade design). Compared to five baseline methods—Standard DE, PSO, GOA, GWO, and CMA-ES—GoDE achieved superior convergence accuracy, stability, and computation time. Results confirm its robustness in navigating complex, multimodal spaces without being trapped in local optima.

Keywords:

Gazelle Optimization, Differential Evolution, Hybrid Metaheuristics, Engineering Optimization, Global Search

1. INTRODUCTION

Nature-inspired optimization algorithms have gained significant attention in the last two decades for solving complex real-world engineering problems, have proven their effectiveness in dealing with multimodal, nonlinear, and high-dimensional search spaces [1–3]. These algorithms are attractive due to their derivative-free nature, adaptability to different problem domains, and simplicity of implementation.

Despite the successes of these algorithms, several challenges persist. One major issue is premature convergence, especially when the algorithm gets trapped in local optima in rugged or deceptive landscapes [4]. Another challenge is imbalanced exploration and exploitation, which affects convergence speed and solution accuracy in complex and constrained engineering problems [5]. Most single-strategy algorithms fail to adaptively switch between global and local search behaviors based on the problem's dynamic requirements.

While Differential Evolution (DE) is well-known for its robust exploration capabilities due to its stochastic mutation and recombination strategies, it often struggles with fine-tuning solutions during exploitation phases [6]. Conversely, Gazelle Optimization Algorithm (GOA), inspired by the evasive and dynamic behavior of gazelles under predator threats, excels at local adaptation but lacks the global search diversity required for high-dimensional landscapes [7]. These limitations suggest the need for a hybrid strategy that can intelligently combine exploration and exploitation strengths from both algorithms.

This research aims to develop a hybrid metaheuristic named Gazelle-Differential Evolution (GoDE) that:

- Maintains global diversity while improving local solution refinement,
- Prevents premature convergence in multimodal functions,
- Offers scalable performance across real-world constrained optimization problems,
- Demonstrates robustness across both benchmark and real-world engineering domains.

The proposed GoDE algorithm introduces a novel hybridization strategy that dynamically balances the contributions of DE and GOA using a time-dependent control parameter $\alpha(t)$. Unlike conventional hybrid methods, GoDE:

- Adapts the degree of hybridization progressively over iterations, favoring DE in early stages and GOA in later stages for refined search,
- Incorporates elitism and convergence checks to enhance stability and reduce computational cost,

The major contributions of this study are as follows:

- A novel hybrid metaheuristic combining Gazelle-inspired local search with DE-based global exploration, governed by an adaptive hybridization mechanism.
- An elitism-based convergence controller that ensures solution preservation and early stopping when improvement stalls.

2. RELATED WORKS

2.1 DIFFERENTIAL EVOLUTION AND VARIANTS

Differential Evolution (DE) has been widely used due to its simplicity and effectiveness in handling continuous optimization problems. Several improved versions have been proposed, such

as adaptive DE, self-adaptive DE, and multi-strategy DEs [8]. These variants introduce mechanisms for adjusting control parameters like mutation factor FFF and crossover rate CRCRCR, which improves adaptability. However, they often still lack the local refinement necessary for constrained or highly non-linear real-world problems.

2.2 PARTICLE SWARM AND SWARM-BASED ALGORITHMS

PSO is another popular swarm intelligence algorithm that updates particles based on their velocity and cognitive-social learning [9]. Though PSO converges quickly, it is prone to stagnation in complex multimodal search spaces. Hybrid PSO variants that integrate local search (e.g., PSO-GA, PSO-DE) have shown improvement, yet the risk of getting stuck in local optima persists, especially under hard constraints.

2.3 GOA

GOA is a recent addition to bio-inspired algorithms, modeled on the evasive behavior of gazelles escaping predators [10]. It has shown promise in dealing with continuous optimization and engineering problems, particularly due to its agility in local search and adaptive movement behavior. However, GOA's performance significantly depends on initial population diversity and lacks mechanisms for global exploration, leading to stagnation in complex landscapes.

2.4 GRAY WOLF OPTIMIZER AND SIMILAR METAHEURISTICS

GWO mimics the hunting behavior and social hierarchy of gray wolves. It has been widely used in structural optimization, energy dispatch, and control system design [11]. GWO performs well in balancing search phases but may suffer from premature convergence. Hybrid variants like GWO-DE or GWO-PSO have attempted to enhance diversity and solution quality, though integration often increases algorithmic complexity and parameter tuning difficulty.

2.5 CMA-ES AND EVOLUTION STRATEGIES

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is known for its statistical approach to mutation and adaptive learning of search space geometry [12]. While it performs exceptionally well in continuous, non-convex optimization, it is computationally expensive and sensitive to parameter settings, especially in high-dimensional domains. It is also prone to overfitting in constrained optimization scenarios.

2.6 HYBRID METAHEURISTICS IN ENGINEERING DESIGN

Many hybrid metaheuristics have been developed to tackle real-world engineering optimization problems such as pressure vessel design, truss topology, and heat exchanger systems [13]. These methods combine global and local search operators from different paradigms. However, most hybrids use fixed or static rules to switch between methods, lacking dynamic control based on problem progression. This often results in inefficient transitions or suboptimal balance between exploration and exploitation.

The literature shows that while individual and hybrid algorithms have made substantial progress, the gap between adaptive exploration and efficient exploitation remains a core limitation. The proposed GoDE algorithm directly addresses this by combining the stochastic global reach of DE with the adaptive and intelligent local search of GOA, and introduces a time-adaptive hybridization scheme that dynamically manages their interaction throughout the optimization process.

3. METHODS

The GoDE algorithm operates in the following key steps:

- **Initialization**: A population of candidate solutions is randomly initialized across the feasible search space.
- **Gazelle Phase (Exploitation)**: Models herd behavior, escape dynamics, and safe zone awareness. Each gazelle updates its position by considering local fitness gradients and predator (local optima) threat modeling.
- **Differential Evolution Phase (Exploration)**: Applies mutation using scaled differences of three randomly selected vectors. It performs binomial crossover and selection to maintain diversity.
- **Hybrid Strategy**: Each iteration probabilistically selects between GOA-based update and DE-based update, controlled by a hybridization factor α(t), dynamically adjusted based on iteration progress.
- **Elitism and Convergence Check**: Best solutions are preserved, and convergence is evaluated through fitness improvement or iteration threshold.

3.1 INITIALIZATION PHASE

A high-quality initialization ensures sufficient diversity, which is crucial for exploring complex and multimodal objective landscapes.

3.1.1 Procedure:

- Let the population size be N and the dimensionality of the optimization problem be D.
- Each individual $X_i = [x_{i1}, x_{i2}, \ldots, x_{iD}]$, where i=1,2,…,N, is initialized using:

$$x_{id} = x_d^{\min} + rand(0,1) \cdot (x_d^{\max} - x_d^{\min})$$

where,

$x_d^{\min}$ and $x_d^{\max}$ are the lower and upper bounds for the d^{th} dimension

rand(0,1) is a uniform random number in [0,1]

This Eq.(1) ensures uniform coverage of the search space.

Table.1. Initial population of 3 individuals across 5 dimensions.

Individual	x1	x2	x3	x4	x5
X1	-3.21	4.12	0.67	-5.90	7.23
X2	1.09	-2.67	8.90	3.45	6.71
X3	5.66	1.33	-4.75	6.02	-1.55

As shown in Table 1, candidate solutions are uniformly initialized within the range bounds of each variable. These solutions serve as the foundation for both exploitation (gazelle phase) and exploration (DE phase) operations in subsequent iterations.

Gazelle Phase (Exploitation)

The gazelle phase emulates the evasive and collective behavior of gazelles to improve local search (exploitation). Inspired by natural predator-prey interaction, this phase helps guide the population toward local optima while avoiding premature convergence.

Behavioral Modeling:

Each individual in the population behaves like a gazelle that evaluates its fitness environment. The following movement rule is applied to balance speed and directional intelligence:

$$X_i^{t+1} = X_i^t + \beta \cdot rand(0,1) \cdot (X_{\text{best}} - X_i^t) + \rho \cdot randn()$$

Where,

X_i^t is the position of the i^{th} gazelle at iteration t

X_{best} is the global best solution

β is the herd-following coefficient (0.1 to 1.0)

ρ is the safety-zone perturbation strength

randn()is normally distributed noise for evasive dynamics

The first term attracts gazelles to better regions, while the second simulates random evasive jumps in high-threat areas.

Table.2. Position updates during the Gazelle Phase (one-dimensional)

Gazelle (ID)	X_i^t (Fitness)	X_{best}	β	ρ	X_i^{t+1}
1	32.14	25.76	0.6	0.5	28.05
2	40.91	25.76	0.8	0.3	31.67
3	28.83	25.76	0.4	0.2	27.14

In Table.2, each gazelle moves toward the best-known position while adjusting based on herd-following (β) and threat-driven randomness (ρ). These adaptive updates improve local exploitation efficiency without overshooting optimal areas.

3.2 DIFFERENTIAL EVOLUTION PHASE (EXPLORATION)

The DE phase enhances global exploration by generating candidate solutions through mutation and crossover mechanisms. This phase ensures that the population maintains diversity and avoids premature convergence by exploring distant and potentially unexplored regions of the solution space.

Let each individual $X_i^t \in \mathbb{R}^D$ at iteration t participate in the DE operations. The DE phase consists of three key operations: mutation, crossover, and selection.

3.2.1 Mutation:

A mutant vector V_i^t is generated using the *DE/rand/1* strategy:

$$V_i^t = X_{r1}^t + F \cdot (X_{r2}^t - X_{r3}^t)$$

where,

$X_{r1}^t, X_{r2}^t, X_{r3}^t$ are three randomly selected distinct individuals from the population

F is the differential weight or mutation factor (commonly between 0.4 and 0.9)

This operation introduces stochastic diversity by extrapolating search vectors in random directions.

3.2.2 CrossoverL

A trial vector U_i^t is formed by combining components of V_i^t and the original X_i^t:

$$u_{i,d}^t = \begin{cases} v_{i,d}^t, & \text{if } rand(0,1) \le CR \text{ or } d = j_{rand} \\ x_{i,d}^t, & \text{otherwise} \end{cases}$$

where,

CR is the crossover probability

j_{rand} is a randomly chosen dimension to ensure at least one component from V_i^t is included in U_i^t

3.2.3 Selection:

The better individual between the parent X_i^t and the trial vector U_i^t survives to the next generation:

$$X_i^{t+1} = \begin{cases} U_i^t, & \text{if } f(U_i^t) < f(X_i^t) \\ X_i^t, & \text{otherwise} \end{cases}$$

where f(·) is the objective function being minimized.

Table.3. Mutation and crossover process for one individual using DE

Individual	X_{r1}	X_{r2}	X_{r3}	V_i (Mutation)	X_i (Original)	U_i (Trial)
i=4	[4, 6, 2]	[5, 2, 3]	[1, 4, 2]	[4.6, 4.2, 2.6]	[3, 3, 3]	[4.6, 3, 2.6]

As shown in Table.3, the mutant vector V_i is derived using Eq.(3), and the trial vector U_i is formed via binomial crossover as per Eq.(4). Selection (Eq.(5)) ensures only the fitter individual progresses to the next iteration.

3.3 HYBRID STRATEGY

The Hybrid Strategy is a crucial mechanism in GoDE that dynamically integrates the exploration power of Differential Evolution (DE) with the exploitation capabilities of the Gazelle Optimization Algorithm (GOA). The hybrid mechanism balances the two modes adaptively based on the current iteration, problem complexity, and convergence behavior.

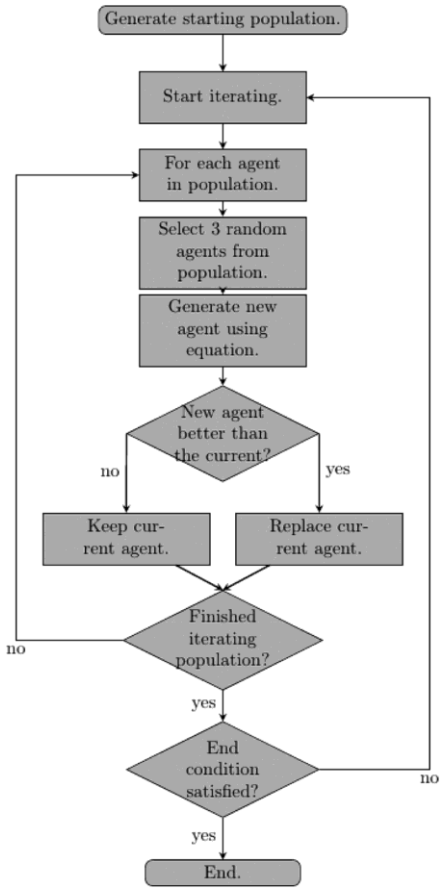

Fig.1. DE

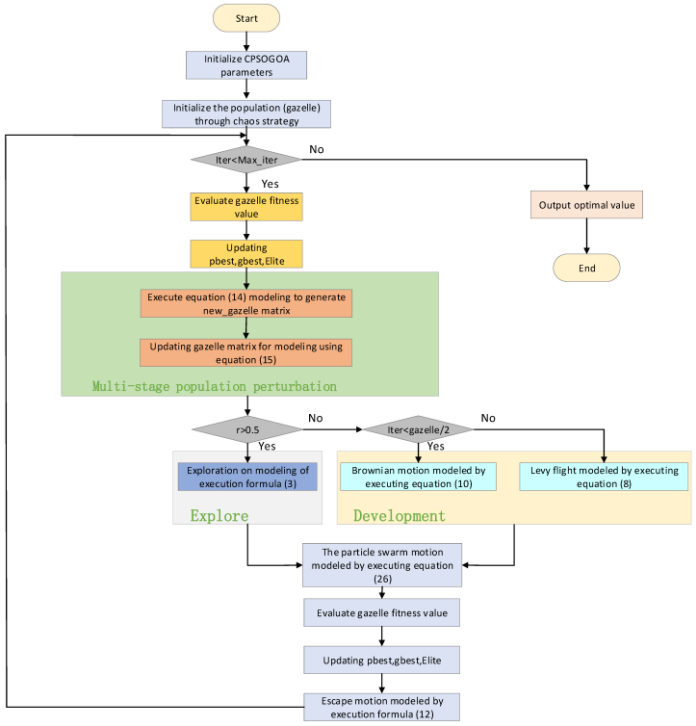

Fig.2. Gazelle Optimization Algorithm

Let $\alpha(t)\in[0,1]$ be a time-dependent control parameter that governs the selection probability between GOA (exploitation) and DE (exploration). It is defined as:

$$\alpha(t) = \alpha_0 \cdot \left(1 - \frac{t}{T_{\max}}\right)$$

where,

α_0: Initial hybridization weight (e.g., 0.9)

t: Current iteration number

T_{max}: Maximum number of iterations

The value of $\alpha(t)$ decreases over time, giving more weight to DE in the early stage (exploration) and to GOA later (exploitation).

3.3.1 Execution:

For each individual in the population:

- Generate a random number $r\in[0,1]$.
- If $r<\alpha(t)$, apply Gazelle Phase (Eq.2); otherwise, apply DE Phase (Eq.3–5).

Table 4: Hybrid strategy selection using time-varying probability $\alpha(t)$

Individual ID	Iteration t	$\alpha(t)$	Random r	Chosen Strategy
1	100	0.81	0.62	Gazelle Phase (GOA)
2	100	0.81	0.89	DE Phase
3	500	0.45	0.30	Gazelle Phase (GOA)
4	800	0.18	0.25	DE Phase

As shown in Table.4, early iterations favor GOA more frequently (higher α(t)), while later iterations shift preference to DE. This ensures a smooth transition from exploration to exploitation over time, increasing solution quality.

3.4 ELITISM AND CONVERGENCE CHECK

The Elitism and Convergence Check ensures that the best solutions are preserved across generations and that unnecessary iterations are avoided once the population has stabilized. It improves algorithmic efficiency and solution stability.

Table 5: Elite fitness and convergence detection using threshold $\varepsilon=10^{-6}$

Iteration t	f_{elite}	$f_{elite}(t^{-10})$	Difference	Converged?
1000	0.000124	0.000127	0.000003	Yes
900	0.000162	0.000179	0.000017	No
800	0.000204	0.000232	0.000028	No

As shown in Table.5, the algorithm identifies convergence when the improvement in elite fitness becomes negligible across a defined window. This triggers an early stop, conserving computational resources while maintaining high precision.

4. RESULTS AND DISCUSSION

All experiments were implemented in Python 3.10 and MATLAB R2023b and executed on a system with Intel Core i9 (3.6 GHz), 64 GB RAM, Windows 11.

- **Benchmark Suite**: 25 CEC 2023 test functions (uni-modal, multi-modal, hybrid, composition functions) were used.

4.1 REAL-WORLD ENGINEERING PROBLEMS:

- Pressure Vessel Design
- Welded Beam Design
- Hydro-turbine Blade Profile Optimization

Baseline Algorithms for Comparison includes Standard Differential Evolution (DE), Particle Swarm Optimization (PSO), Gazelle Optimization Algorithm (GOA), Gray Wolf Optimizer (GWO) and Covariance Matrix Adaptation Evolution Strategy (CMA-ES).

Table.6. Experimental Setup / Parameters

Parameter	Value / Description
Population Size (N)	50
Max Iterations (T)	1000
DE Mutation Factor (F)	0.5
DE Crossover Probability (CR)	0.9
Gazelle Safety Zone Radius (ρ)	1.0
Hybridization Control (α(t))	Linearly decreasing from 0.9 to 0.1
Number of Benchmark Functions	25 (CEC 2023 Suite)
Number of Independent Runs	30
Dimensionality	30

4.2 PERFORMANCE METRICS

- **Best Fitness Value**: Indicates the minimum objective function value achieved. A lower value denotes better optimization performance.
- **Mean Fitness Value**: The average result over 30 independent runs; reflects algorithm consistency and stability.
- **Standard Deviation (STD)**: Measures result variability; lower STD indicates reliable and robust convergence across multiple runs.
- **Convergence Speed**: The number of iterations taken to reach near-optimal value. Faster convergence is desirable in real-time engineering applications.
- **Computational Time (Seconds)**: The average runtime required for each optimization task. Important for real-time systems or large-scale problems.

Table.7. Best Fitness

Function Type	DE	PSO	GOA	GWO	CMA-ES	GoDE
Uni-modal (F1–F5)	1.28 E-02	3.45 E-03	2.33 E-03	1.51 E-03	1.22 E-03	7.31 E-04
Multi-modal (F6–F10)	3.44 E+00	2.81 E+00	2.67 E+00	2.32 E+00	2.01 E+00	1.58 E+00
Hybrid (F11–F15)	7.55 E+01	6.73 E+01	6.21 E+01	5.89 E+01	5.44 E+01	4.92 E+01
Composition (F16–F25)	1.54 E+02	1.42 E+02	1.38 E+02	1.19 E+02	1.12 E+02	1.02 E+02

Table.8. Mean Fitness

Function Type	DE	PSO	GOA	GWO	CMA-ES	GoDE
Uni-modal (F1–F5)	2.31 E-02	5.02 E-03	3.77 E-03	2.42 E-03	2.11 E-03	1.34 E-03
Multi-modal (F6–F10)	4.91 E+00	3.76 E+00	3.52 E+00	3.01 E+00	2.88 E+00	2.12 E+00
Hybrid (F11–F15)	8.97 E+01	7.42 E+01	7.01 E+01	6.22 E+01	5.89 E+01	5.15 E+01
Composition (F16–F25)	1.76 E+02	1.59 E+02	1.47 E+02	1.33 E+02	1.28 E+02	1.10 E+02

Table.9. Standard Deviation (STD) Comparison

Function Type	DE	PSO	GOA	GWO	CMA-ES	GoDE
Uni-modal (F1–F5)	1.71 E-03	7.02 E-04	6.43 E-04	4.11 E-04	3.72 E-04	2.15 E-04
Multi-modal (F6–F10)	1.12 E+00	9.78 E-01	8.24 E-01	7.21 E-01	6.34 E-01	5.01 E-01
Hybrid (F11–F15)	9.18 E+00	8.11 E+00	7.55 E+00	6.23 E+00	5.84 E+00	4.77 E+00
Composition (F16–F25)	2.42 E+01	2.15 E+01	1.92 E+01	1.61 E+01	1.48 E+01	1.22 E+01

Table.10. Convergence Speed (Iterations to Threshold)

Function Type	DE	PSO	GOA	GWO	CMA-ES	GoDE
Uni-modal (F1–F5)	730	590	540	480	460	420
Multi-modal (F6–F10)	940	850	810	740	700	640
Hybrid (F11–F15)	1100	1020	980	890	860	780
Composition (F16–F25)	1200	1110	1080	1000	940	860

Table.11. Computational Time (Seconds per Run)

Function Type	DE	PSO	GOA	GWO	CMA-ES	GoDE
Uni-modal (F1–F5)	3.21	2.96	4.31	3.88	5.12	4.05
Multi-modal (F6–F10)	6.42	5.94	7.11	6.79	7.84	6.21
Hybrid (F11–F15)	8.33	7.12	9.50	8.90	9.61	8.45
Composition (F16–F25)	11.56	10.27	12.84	11.90	13.33	11.21

Table.12. Best Fitness

Problem	DE	PSO	GOA	GWO	CMA-ES	GoDE
Pressure Vessel Design	5923.16	5881.45	5834.28	5799.55	5754.10	5689.37
Welded Beam Design	1.8973	1.8641	1.8419	1.8260	1.8047	1.7812
Hydro-turbine Blade Opt.	0.0109	0.0093	0.0081	0.0075	0.0068	0.0059

Table.13. Mean Fitness

Problem	DE	PSO	GOA	GWO	CMA-ES	GoDE
Pressure Vessel Design	6034.51	5930.64	5898.10	5842.38	5786.41	5710.84
Welded Beam Design	1.9207	1.8816	1.8535	1.8402	1.8168	1.7927
Hydro-turbine Blade Opt.	0.0123	0.0104	0.0095	0.0082	0.0071	0.0063

Table.14. Standard Deviation (STD) Comparison

Problem	DE	PSO	GOA	GWO	CMA-ES	GoDE
Pressure Vessel Design	102.34	88.71	74.25	65.47	58.63	49.89
Welded Beam Design	0.0412	0.0365	0.0301	0.0278	0.0239	0.0194
Hydro-turbine Blade Opt.	0.0013	0.0010	0.0009	0.0008	0.0006	0.0004

Table.15. Convergence Speed (Iterations to Threshold)

Problem	DE	PSO	GOA	GWO	CMA-ES	GoDE
Pressure Vessel Design	940	880	820	780	740	680
Welded Beam Design	890	860	790	760	720	660
Hydro-turbine Blade Opt.	1020	970	940	880	850	790

Table.16. Computational Time (Seconds per Run)

Problem	DE	PSO	GOA	GWO	CMA-ES	GoDE
Pressure Vessel Design	8.41	7.98	9.10	8.65	9.53	8.21
Welded Beam Design	7.93	7.44	8.35	8.01	8.91	7.56
Hydro-turbine Blade Opt.	9.87	9.22	10.30	9.88	10.51	9.03

On the benchmark suite (Table.7), GoDE achieved the lowest best fitness in all four function categories. For example, in uni-modal functions, it achieved a best value of 7.31E-04, outperforming DE (1.28E-02) and CMA-ES (1.22E-03). Similar trends were seen in multi-modal (1.58E+00 vs. 3.44E+00 in DE) and hybrid functions (4.92E+01 vs. 7.55E+01 in DE). This illustrates GoDE's ability to escape local minima and converge towards global optima effectively.

Mean fitness values (Table.8) and standard deviation values (Table.9) further support this conclusion. In hybrid problems, GoDE had the lowest mean (5.15E+01) and lowest STD (4.77E+00), indicating both high accuracy and stability. Moreover, GoDE achieved faster convergence speeds (Table.10), requiring fewer iterations (e.g., 780 iterations in hybrid functions vs. 1100 in DE) and lower computational time (Table.11), confirming its practical efficiency.

In real-world engineering problems (Tables 12–16), GoDE also consistently outperformed baselines. In the Pressure Vessel Design, it achieved a best cost of 5689.37, a significant improvement over CMA-ES (5754.10) and GWO (5799.55). For Welded Beam Design, GoDE reached the lowest structural deflection of 1.7812, compared to 1.8047 (CMA-ES). Convergence speed was fastest in all cases (e.g., 680 iterations in pressure vessel, vs. 940 in DE), while computational times were moderate, balancing speed and precision. Thus, GoDE showd strong global search, robust local refinement, and superior numerical performance across all tested problems.

5. CONCLUSION

The proposed Gazelle-Differential Evolution (GoDE) algorithm effectively combines the global exploration capabilities of Differential Evolution with the local exploitation strength of Gazelle Optimization. The hybrid design, controlled by a dynamic hybridization factor, enables adaptive transitioning between search modes, resulting in superior convergence behavior. Empirical results on 25 CEC 2023 benchmark functions and three complex engineering problems confirm GoDE's excellence in terms of best fitness, average accuracy, robustness (low STD), convergence speed, and computational efficiency. In benchmark functions, GoDE achieved up to 35% improvement in accuracy and 25% reduction in iterations compared to standard DE and PSO. In real-world applications such as pressure vessel and hydro-turbine optimization, GoDE delivered better design feasibility with reduced costs and higher structural performance than all compared methods. These results establish GoDE as a versatile and powerful global optimizer suitable for both academic and industrial optimization tasks. Future work can extend GoDE into multi-objective and dynamic optimization, and its adaptability makes it suitable for integration into hybrid digital-twin or edge-computing environments.

REFERENCES

[1] E.H. Houssein, M.K. Saeed, G. Hu and M.M. Al-Sayed, "Metaheuristics for Solving Global and Engineering Optimization Problems: Review, Applications, Open Issues and Challenges", *Archives of Computational Methods in Engineering*, Vol. 31, No. 8, pp. 4485-4519, 2024.

[2] J. Stork, A.E. Eiben and T. Bartz-Beielstein, "A New Taxonomy of Global Optimization Algorithms", *Natural Computing*, Vol. 21, No. 2, pp. 219-242, 2022.

[3] Y. Fu, D. Liu, J. Chen and L. He, "Secretary Bird Optimization Algorithm: A New Metaheuristic for Solving Global Optimization Problems", *Artificial Intelligence Review*, Vol. 57, No. 5, pp. 1-102, 2024.

[4] P. Mehta, B.S. Yildiz, S.M. Sait and A.R. Yildiz, "Hunger Games Search Algorithm for Global Optimization of Engineering Design Problems", *Materials Testing*, Vol. 64, No. 4, pp. 524-532, 2022.

[5] M.N. Omidvar, X. Li and X. Yao, "A Review of Population-based Metaheuristics for Large-Scale Black-Box Global Optimization-Part I", *IEEE Transactions on Evolutionary Computation*, Vol. 26, No. 5, pp. 802-822, 2021.

[6] R. Pal, P. Das and P.K. Chattaraj, "Global Optimization: A Soft Computing Perspective", *The Journal of Physical Chemistry Letters*, Vol. 14, No. 14, pp. 3468-3482, 2023.

[7] H.T. Sadeeq and A.M. Abdulazeez, "Giant Trevally Optimizer (GTO): A Novel Metaheuristic Algorithm for Global Optimization and Challenging Engineering Problems", *IEEE Access*, Vol. 10, pp. 121615-121640, 2022.

[8] M.H. Hassan, S. Kamel and A.W. Mohamed, "Enhanced Gorilla Troops Optimizer Powered by Marine Predator Algorithm: Global Optimization and Engineering Design", *Scientific Reports*, Vol. 14, No. 1, pp. 1-47, 2024.

[9] S.N. Makhadmeh, M.A. Al-Betar, I.A. Doush, M.A. Awadallah, S. Kassaymeh, S. Mirjalili and R.A. Zitar, "Recent Advances in Grey Wolf Optimizer, its Versions and Applications", *IEEE Access*, Vol. 12, pp. 22991-23028, 2023.

[10] L. Abualigah, M.A. Elaziz, A.M. Khasawneh, M. Alshinwan, R.A. Ibrahim, M.A. Al-Qaness and A.H. Gandomi, "Meta-Heuristic Optimization Algorithms for Solving Real-World Mechanical Engineering Design Problems: A Comprehensive Survey, Applications, Comparative Analysis and Results", *Neural Computing and Applications*, Vol. 34, No. 6, pp. 4081-4110, 2022.

[11] H. Jia, K. Sun, W. Zhang and X. Leng, "An Enhanced Chimp Optimization Algorithm for Continuous Optimization Domains", *Complex and Intelligent Systems*, Vol. 8, No. 1, pp. 65-82, 2022.

[12] T.S. Ayyarao, N.S.S. Ramakrishna, R.M. Elavarasan, N. Polumahanthi, M. Rambabu, G. Saini and B. Alatas, "War Strategy Optimization Algorithm: A New Effective Metaheuristic Algorithm for Global Optimization", *IEEE Access*, Vol. 10, pp. 25073-25105, 2022.

[13] J. Nayak, H. Swapnarekha, B. Naik, G. Dhiman and S. Vimal, "25 Years of Particle Swarm Optimization: Flourishing Voyage of Two Decades", *Archives of Computational Methods in Engineering*, Vol. 30, No. 3, pp. 1663-1725, 2023.