International Journal for Housing Science and Its Applications
Publish August 10, 2025. Volume 46, Issue 4 Pages 3522-3533

https://doi.org/10.70517/ijhsa464481a

Interior Layout Design Based on Interactive Differential Evolution Algorithm and Reverse Learning Strategy

Mengna Huang[1,*]
[1] Changzhou University, School of Art and Design, Changzhou, 213000, China
Corresponding authors: (e-mail: 18351221665@163.com).

Abstract The traditional interior layout design method has the problem that it is difficult to express the optimization goal clearly, and the design process lacks flexibility and individualization performance. To solve the above problems, interactive differential evolution algorithm and reverse learning strategy are used to optimize interior layout design, so as to better meet users' individual needs for interior layout design. The study first analyzes the indoor layout design based on improved interactive differential evolution algorithm, then analyzes the application of interactive differential evolution indoor layout with reverse learning strategy, and finally analyzes the performance and application results of the interactive differential algorithm with reverse learning strategy. It was verified that the algorithm had the fastest running time in the unimodal function F1, which was about 14 seconds. In addition, the algorithm could find the global optimum in the four benchmark functions F1, F5, F7, and F10. For European style space design, the research model had a high level of user satisfaction, with the highest satisfaction value reaching 88 in the later stages of iteration. For minimalist style space design, the research model had a high level of user satisfaction, with the highest satisfaction value reaching 98 in the later stages of iteration . The study demonstrates that the integration of an interactive differential evolution algorithm and reverse learning strategy enhances the flexibility and personalization of interior layout design. This approach substantiates the research method's potential and advantages in the domain of interior space layout design, offering novel insights and methods for future research.

Index Terms Interactive, Differential Evolution Algorithm, Reverse learning strategy, Subpopulation strategy, Interior layout design

I. Introduction

The interior layout design industry has shown a rapid development trend worldwide currently [1]. The requirements for the aesthetics and functionality of indoor space layout are increasing. This provides abroad market opportunity for the indoor space layout design industry, but also puts forward higher standards and unprecedented challenges for the industry [2]. Nowadays, various optimization algorithms have been applied in architectural design to assist designers in more accurate and efficient layout design. Differential Evolution (DE) is an emerging and efficient evolutionary computation technique, which has simple structure, strong global search capability, and high robustness [3]. Niu et al. introduced DE algorithm into the field of aircraft manufacturing, the classification performance of minority data was improved, bringing innovation to the parameter optimization, material selection and cost control of aircraft design [4]. As an additional illustration, in the field of power generation, Kar et al. devised a viable configuration for flexible AC transmission system controllers through the application of a DE algorithm. In addition to directly optimizing power grid operation, this study may also provide more robust technical support for the construction of smart grids, the distribution of energy, and the prediction of faults [5]. The DE algorithm simulates a stochastic model of biological evolution, performs random searches in continuous space [6]. However, traditional DE algorithms still have some shortcomings in solving complex optimization, such as premature convergence and low search efficiency. To further improve the performance of the algorithm, researchers have proposed various improvement strategies, among which the Position Based Learning (OBL) strategy, as an effective global search strategy, has been widely used in various optimization algorithms [7]. Scholar Si proposed a firefly swarm optimization algorithm by combining Reverse Learning Strategy (RLS), and applied the improved GSO to the registration of pre - and post-treatment MR images of brain tumor progression. The algorithm performed well in brain MRI registration , it has brought new breakthroughs in the fields of medical image analysis, disease diagnosis and treatment effect evaluation [8]. OBL can accelerate the convergence rate of the algorithm and improve the quality of the solution by considering the information of the solution space and its inverse space simultaneously. Despite significant advancements in the research of DE

algorithms and reverse learning strategies, numerous challenges remain to be addressed in the specific domain of interior layout design. For example, scholar Si cannot reasonably define the reverse space and effectively use the information of the reverse space to guide the search process of the algorithm. In addition, how to meet the individual needs of users while maintaining the overall aesthetics and functionality of the design, and how to achieve efficient user interaction in the design process, so that designers can adjust and optimize the layout in real time, are also important topics in the current interior layout design research. Therefore, in light of the findings of previous research, this study put forth a methodology for interior layout design that integrates the enhanced DE algorithm and the improved OBL algorithm. This approach is designed to address the personalized interior layout design needs of users. Compared with previous studies, this study integrates the global search advantages of OBL strategy, thus effectively avoiding the problems of premature convergence and low search efficiency. At the same time, it pays attention to the flexibility of user interaction, and introduces the reverse learning strategy. It can consider the information of the solution space and its reverse space at the same time, so as to find a better layout design scheme. It is evident that the research method can not only surmount the constraints of conventional DE algorithms in addressing intricate optimization challenges but also prioritize the adaptability of user engagement and the extent of design differentiation, thereby enabling designers to refine and optimize the layout in real-time throughout the design process, and consequently enhance the degree of design differentiation and practicality.

II. Methods And Materials

II. A.Interior Layout Design Based on Improved DE Algorithm

In the process of designing the spatial layout of a house, it is necessary to first establish the overall framework outline ofthe layout. This is followed by the planning of the internal spatial areas, which must be done in accordance with the personalized needs of the family. Evolutionary algorithms can automatically adjust and optimize design schemes to adapt to various changes that may occur during the family lifecycle by simulating natural selection and genetic mechanisms. In evolutionary algorithms, the encoding of chromosomes has evolved into diverse forms to meet the needs of different optimization problems [9], [10]. This study adopts a binary encoding scheme to cleverly map the distribution space parameters to be optimized into a series of binary bit strings, efficiently representing and optimizing complex parameter spaces. The detailed diagram of chromosome coding is shown in Figure 1.

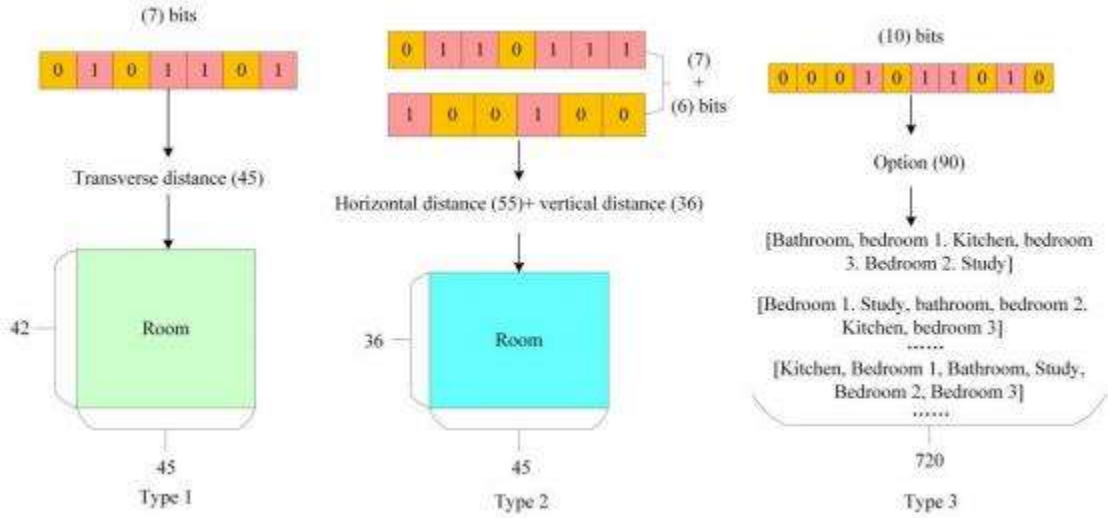

Figure 1: Detailed Chromosomal Coding Diagram

In Figure 1, each chromosome is a sequence consisting of 0s and 1s, and the length of the sequence depends on the accuracy and complexity of the representation required. The DE algorithm is a population-based heuristic search algorithm, wherein each member of the group is represented by a solution vector [11]. The evolutionary process of the DE algorithm encompasses three distinct operations: mutation, hybridization, and selection. However, these operations have varying meanings within the context of evolutionary algorithms. The DE algorithm generates a new individual by combining the vector difference between any two individuals in the initial population and a third individual. This new individual is then compared to the initial population [12]. In the search space, the individual expression of the initial population is shown in equation (1).

$$Xi\ (t) = \left[Xi,1(t),\ Xi,2(t),\ Xi,3(t),\ ...,\ Xi\ ,N\ (t) \right] i = 1,\ 2,3,\ ...,\ NP \quad (1)$$

In equation (1), individuals in the initial population are represented by symbols i . Population size is represented by symbols NP , and population evolution times are represented by symbols t .

$$Xi\ ,j\ (t) = Lmin + rand(0,1)(Lmax - Lmin\ )i = 1,\ 2,3,...,\ NP\ \ j = 1,\ 2,3,\ ...,\ n \quad (2)$$

In equation (2), the size of the search space dimension is denoted by symbol n , and the components are denoted by symbol j , where rand(0,1) is randomly generated data that follows a uniform distribution within the range of (0,1). The upper limit of the search space is represented by a symbol Lmax . The lower limit of the search space is represented by a symbol Lmin . The crossover operation of the DE algorithm involves randomly selecting genes from mutated individuals and target individuals to form a new individual. The expression for the crossover operation of the DE algorithm is shown in equation (3).

$$U_{i,j}(t) = \begin{cases} H_{i,j}(t), rand(0,1) < CR, j = j_{rand} \\ X_{i,j}(t), else \end{cases} \quad (3) \quad (3)$$

In equation (3), symbol CR represents the crossover probability. jrand represents a random integer within the range of [1, n] values. The selection operation of the DE algorithm is based on a greedy strategy, which directly compares the target individual with the mutated individual and selects the better individual to enter the next generation. The selection operation expression of the DE algorithm is shown in equation (4).

$$X_i(t+1) = \begin{cases} U_i(t), f(U_i(t)) < f(X_i(t)) \\ X_i(t), else \end{cases} \quad (4) \quad (4)$$

In equation (4), f (Xi (t)) represents the t -th target individual in the i -th generation. f (Ui (t)) represents the fitness value of the i -th experimental individual. Interactive Differential Evolution (IDE) is a kind of technology that uses the user's perceptual information to optimize and retrieve candidate solutions. It integrates human knowledge, experience and preferences into the optimization process. A human evaluation-based system is used to search for solutions [13]. The essence of IDE is the improvement of DE algorithm, Make use of the global optimization capability of DE algorithm and introduce human subjective intervention [14]. The most important difference between the two is that the IDE algorithm does not need to calculate the fitness value of the evolved individual, but changes the fitness value function to be compared by the user's subjectivity, and the user chooses the individual that is more in line with his favorite, that is, the comparison of the individual fitness value is completed. A schematic diagram of the IDE algorithm flow is provided in Figure 2 for reference.

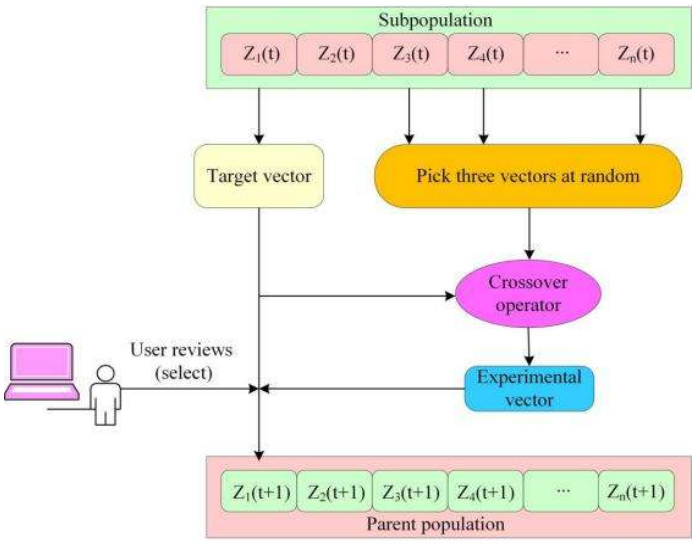

Figure 2: IDE Algorithm Flow Diagram

The specific expression for population initialization of IDE algorithm is shown in equation (5).

$$X0i\ ,j = a + (b - a) \bullet rand(0,1) \quad (5)$$

In equation (5), a denotes the lower bound. b represents the upper bound. The IDE algorithm combines the parameters of the mutation vector with those of a pre-specified target vector in accordance with defined rules to

generate sub-individuals. Assuming that in the g -th iteration, the generated individuals are X1 (g) , X2 (g) , X3 (g) , and the expression of the mutation variable H(g) is shown in equation (6)).

$$H(g) = X1(g) + F \cdot (X2(g) - X3(g)) \quad (6)$$

In equation (6), F represents the scaling factor used to control the step size of mutation. Newly generated offspring are only replaced when they are better than the target individuals in the population. Although the IDE algorithm has fewer parameters and high computational efficiency, it inevitably falls into local optima. Therefore, this study introduces Backtracking Strategy (BS) to optimize the IDE algorithm. The iterative process of the IDE algorithm with BS is shown in Figure 3.

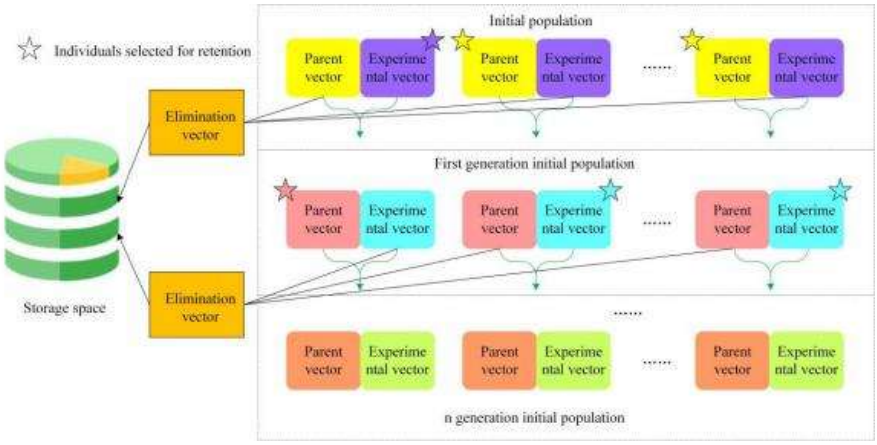

Figure 3: Iterative Process Diagram of IDE Algorithm with BS

In Figure 3, the Euclidean distance information between evolved individuals in the target feature vector can assist users in the simulation evaluation process and provide intuitive reference. The distance calculation expression is shown in equation (7).

$$f(x) = \sqrt{\sum_{i=1}^{n} (x_i - o_i)^2} \quad (7) \quad (7)$$

Equation (7), o represents the target individual.

II. B.IDE Indoor Layout Application with RLS

In evolutionary algorithms, the search starts from a random point in space and may converge to the optimal solution after several iterations. Therefore, OBL strategy is applied to improve the IDE algorithm based on BS [15]. OBL strategy is a method to improve the performance of traditional optimization algorithms by introducing opposition elements. Its core idea is to introduce opposition solutions, , into the search space to improve the diversity and global search [16]. Assuming there exists a point Z in an n-dimensional coordinate system that satisfies X ∈ [LB, UB], the OBL strategy calculation equation is shown in equation (8).

$$\bar{Z} = LB + UB - Z \quad (8)$$

In equation (8), Z is the current solution, and $\bar{Z}$ represents the reverse solution. Random Opposition based Leaming (ROBL) refers to the use of randomness to explore possible action plans and evaluate the effectiveness of these actions through simulation or actual experience, to adjust the strategy to achieve the optimal or better solution [17]. The expression for ROBLis shown in equation (9).

$$Z_{rand} = LB + UB - r - Z \quad (9) \quad (9)$$

In equation (9), r denotes a random number within [0,1]. The quasi RLS (Quasi OBL, QOBL) does not directly take the symmetric point of the current solution in the search space when generating the reverse solution, but takes a random point located between the current solution and the boundary of the search space as the reverse solution [18]. The calculation equation for QOBLis shown in equation (10).

$$\tilde{Z}_j^{qo} = \begin{cases} rand(M_j, \tilde{Z}_j), Z_J < M_j \\ rand(\tilde{Z}_j, M_j), otherwise \end{cases} \quad (10) \quad (10)$$

In equation (10), j denotes the midpoint of the j-th dimensional variable. The expression for j is shown in

equation (11).

$$M_j = \frac{a_j + b_j}{2} \tag{11}$$

Current Optimal Opposition-Based Learning (COOBL) refers to the use of the information of the current optimal solution by reverse thinking or changing some features or parameters of the current optimal solution. to explore new possible solutions in the solution space [19]. The expression of the policy is shown in equation (12).

$$Z\tilde{j}(co) = 2z\text{best},j - Zj \tag{12}$$

Generalized RLS (GOBL) is developed on the basis of OBL, which refers to the process in which learners engage in reverse thinking about their own learning activities, as well as the things, materials, information, thinking, and results involved in the process. The calculation equation for GOBLis shown in equation (13).

$$Z\tilde{j}(go) = k.(aj + bj) - Zj \tag{13}$$

In equation (13), k denotes a random number distributed within [0,1]. The center based RLS (Centroid OBL, COBL) mainly generates reverse solutions by utilizing the center points of individuals in the population, thereby guiding the search process towards a direction that is more likely to find the global optimal solution [20]. The calculation equation for COBLis shown in equation (14).

$$\begin{matrix} Z\,co = 2C - Z \\ j \quad j \quad j \\ C \qquad\qquad C \end{matrix} \tag{14}$$

In equation (14), j represents the center point. The expression for j is shown in equation (15).

$$C_j = \frac{\sum_{i=1}^{NP} Z_{i,j}}{NP} \tag{15}$$

The method of jump probability refers to the process of using random number generation and function value calculation. This method is based on the random search method, which generates multiple sets of random numbers evenly within [0,1] [21]. Then use these random numbers to construct points within the hypercube and calculate the function values of these points. By generating a large number of points and calculating the objective function values of the corresponding points, the point that minimizes the objective function value is finally selected as the required minimum point. The research utilizes this method to generate subpopulations, and the operation process is shown in Figure 4.

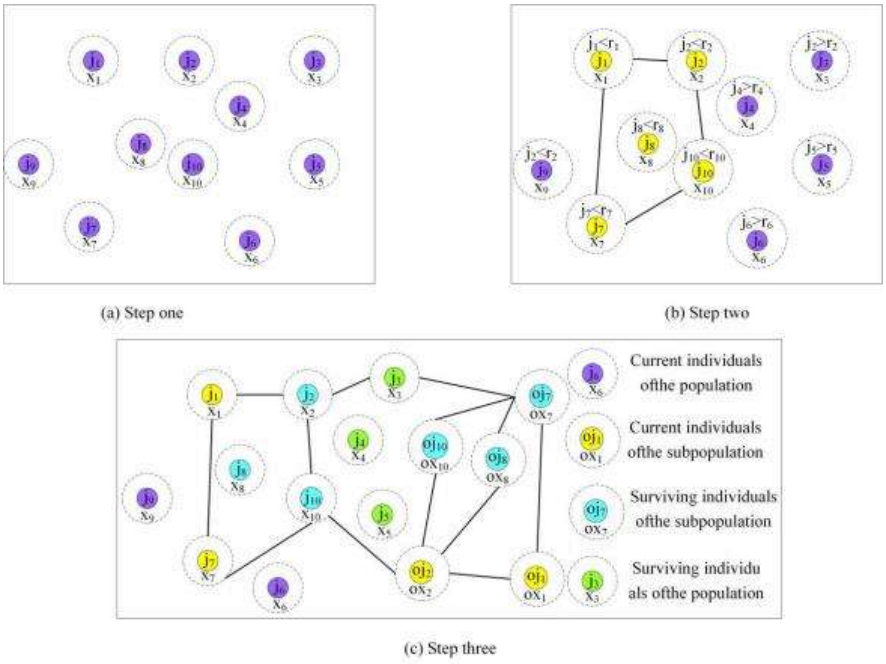

Figure 4: Reverse Learning IDE Algorithm with Subpopulation Strategy

Figure 4 (a) shows the first step of the reverse learning IDE algorithm with subpopulation strategy. After population initialization, the jump probability j corresponding to individuals is randomly generated according to

Gaussian distribution. Figure 4 (b) shows the second step of the reverse learning IDE algorithm with subpopulation strategy, where a random number r is generated for each individual and the corresponding jump probability j is compared. Figure 4 (c) shows the third step of the reverse learning IDE algorithm based on subpopulation strategy. To further optimize the integration of the reverse learning IDE algorithm framework with practical application scenarios, a new indoor layout design scheme has been proposed. This scheme introduces genetic encoding with the aim of efficiently exploring and optimizing the layout and configuration of indoor spaces through the principles of genetic algorithms. The new scheme design process is shown in Figure 5.

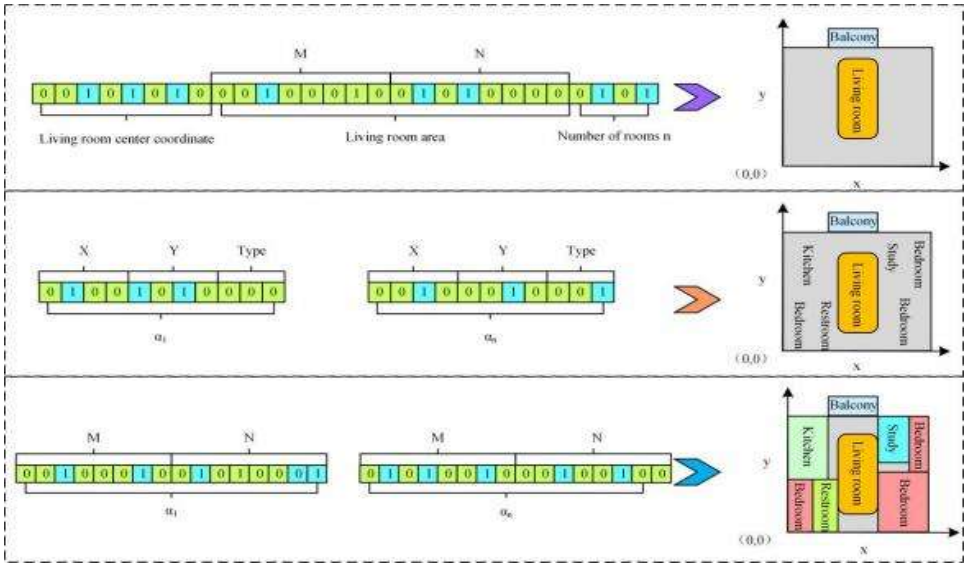

Figure 5: Interior Layout Design Based on Genetic Coding

In Figure 5, the genetic coding of indoor layout design can be divided into three parts. The first part is the smallest gene unit, which is the center coordinate within the preset area. Using the generated center point as the origin, it radiates outward to generate M and N coordinate axes, determining the boundary range of the living room. The second part is the dynamic generation of the center coordinates of the functional area. Based on the total number of rooms n determined in the first part, this part adopts a cyclic iteration method to dynamically generate the center coordinate points of each functional area on the house model. The third part is the intelligent determination of wall boundaries. On the basis of the first two parts, this part starts from the center point of the functional area and intelligently extends outwards to determine the M and N coordinate points one by one, thus determining the wall contour of each functional area. The IDE indoor layout design process based on RLS is shown in Figure 6.

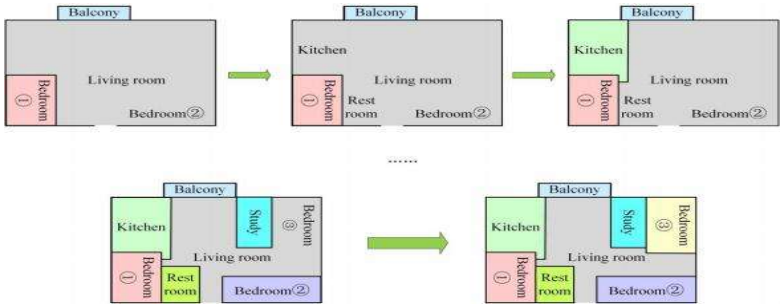

Figure 6: IDE Interior Layout Design Process Based on RLS

As shown in Figure 6, a simplified strategy is adopted for the initial positioning of the room, which ensures that it is located in a reasonable position within the overall framework of the house. This process starts with the preset center point of the room and then uses the standard rectangular area as a reference to accurately define the room's wall boundaries by intelligently expanding into the surrounding areas. Once the room layout is complete, its detailed wall position information is immediately fed back into the subsequent design stage as a key constraint

condition for planning the room. When conceptualizing the layout of a room, the system takes into account the overall building framework and the space already occupied by the room, to avoid spatial overlap and ensure the rationality of the design. This process will serve as a loop mechanism, applied one by one to optimize the wall layout of each room, until all rooms in the entire indoor space are efficiently and accurately partitioned.

III. Results

III. A. Performance Testing and Analysis of Interactive Differential Algorithm Based on RLS

To test the performance superiority of the research method, a comparative experiment was conducted between the IDE-BO-OBL algorithm and two state-of-the-art OBL variants, COBL and QOBL. The study used 10 benchmark functions to explain the performance of the algorithm and selected CEC2017 as the testing function for this experiment. In the CEC2017 test function, F1 and F2 were unimodal functions, F3, F4, and F5 were multimodal functions, F6 and F7 were mixed functions, and F8, F9, and F10 were combination functions. The comparison of the running times of various

algorithms was presented in Figure 7.

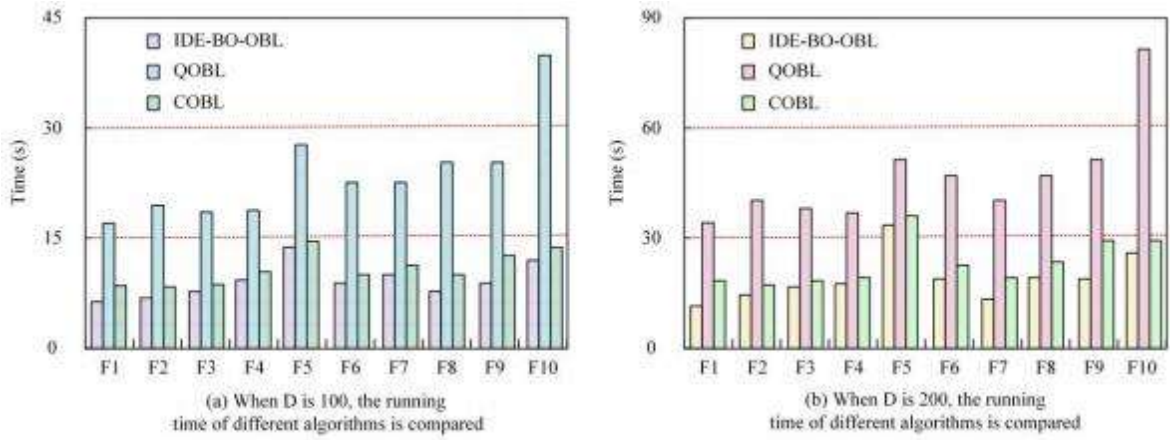

Figure 7: Comparison Chart of the Running Time of Different Algorithms

Figure 7 (a) showed a comparison of the running time of three algorithms when the dimension was 100. In Figure 7 (a), under the condition of 100 dimensions, the three algorithms had the fastest running time in the unimodal function F1. The IDE-BO-OBL algorithm had a running time of about 7 seconds, the QOBL algorithm had a running time of about 17 seconds, and the COBL algorithm had a running time of about 10 seconds. In the combination function F10, the three algorithms had the slowest running time, with the IDE-BO-OBL algorithm running for about 13 seconds, the COBLQOBL algorithm running for about 42 seconds, and the COBL algorithm running for about 14 seconds. Figure 7 (b) showed a comparison of the running time of three algorithms when the dimension was 200. According to Figure 7 (b), under the condition of 200 dimensions, the three algorithms had the fastest running time in the unimodal function F1. The running time of IDE-BO-OBL algorithm was about 14 seconds, COBLQOBL algorithm was about 31 seconds, and COBL algorithm was about 17 seconds. The three algorithms had the slowest running time in the combination function F10. The running time of IDE-BO-OBL algorithm was about 28 seconds, COBLQOBL algorithm was about 88 seconds, and COBL algorithm was about 29 seconds. From this, as the dimension increases, the running time of all three types of algorithms increased, and the running time of IDE-BO-OBL algorithm was more advantageous compared to the two advanced OBL variants. The convergence curves of different algorithms on F1, F5, F7, and F10 were shown in Figure 8.

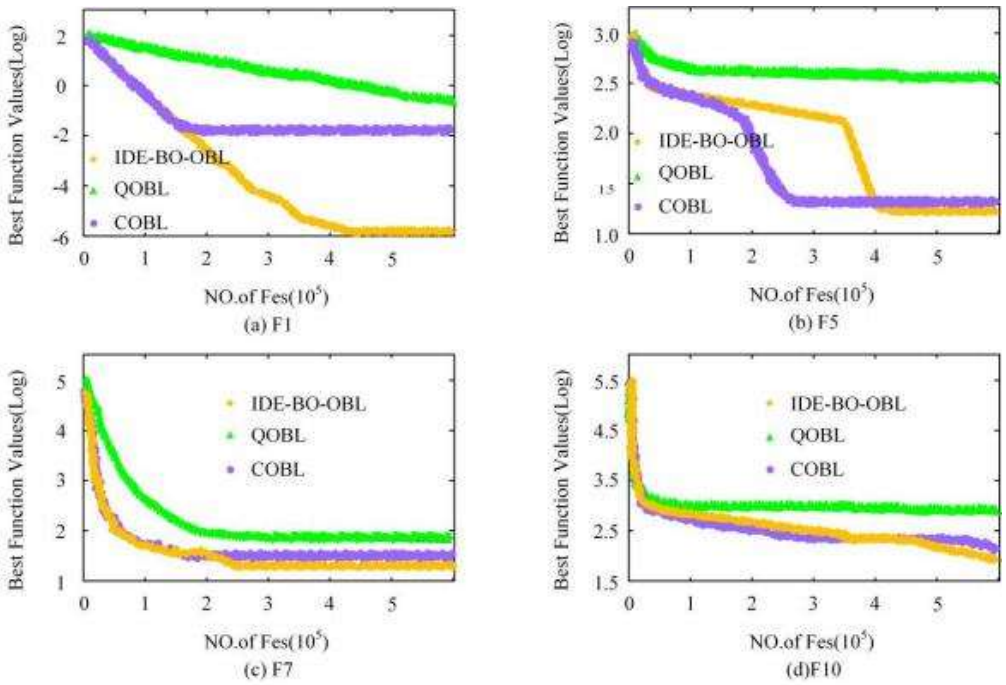

Figure 8: Convergence Graphs of Different Algorithms on F1, F5, F7 and F10

In Figure 8 (a), the IDE-BO-OBL algorithm found the global optimum in F1, while the QOBL and COBL algorithms fell into the local optimum. In Figure 8 (b), the IDE-BO-OBL algorithm and COBL algorithm found the global optimum in F5, while the QOBL algorithm remained the local optimum. In Figure 8 (c), the IDE-BO-OBL algorithm and COBL algorithm converged faster than the QOBL algorithm at the beginning of iteration. In Figure 8 (d), all three algorithms reached the global optimum in F10, and the IDE-BO-OBL algorithm had the fastest convergence speed. The IDE-BO-OBL algorithm could find the global optimal state in different types of benchmark functions and has strong applicability. The boxplots of IDE-BO-OBL algorithm at different p-values were shown in Figure 9.

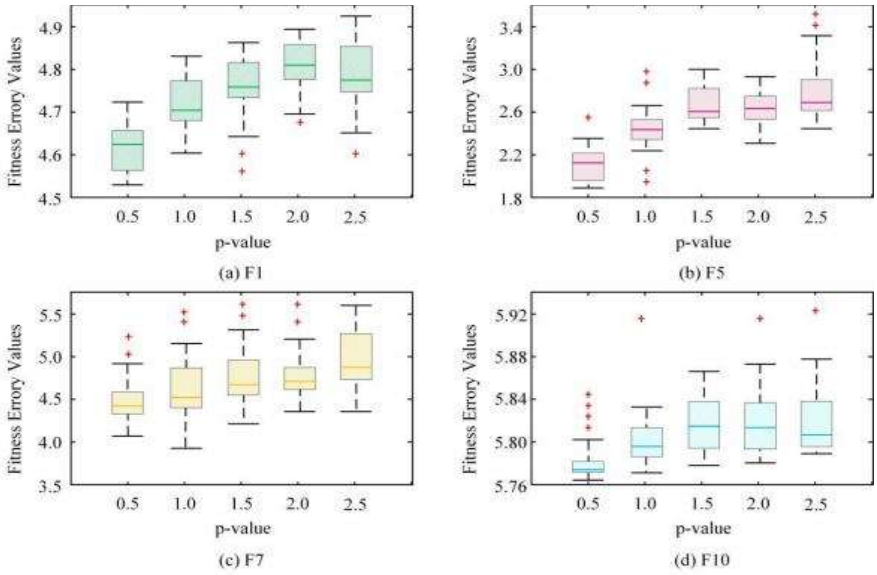

Figure 9: Box Plot of IDE-BO-OBL Algorithm Under Different P-values

Figure 9 (a) showed the box plots of the IDE-BO-OBL algorithm in F1 at different p-values. In Figure 9 (a), when the p-value was 0.5, the minimum fitness error value of the IDE-BO-OBL algorithm was 4.52. When the p-value was 2.5, the maximum fitness error value of the IDE-BO-OBL algorithm was 4.95. Figure 9 (b) showed the box

plot of the IDE-BO-OBL algorithm in F5 at different p-values. In Figure 9 (b), when the p-value was 0.5, the minimum fitness error value of the IDE-BO-OBL algorithm was 1.96. When the p-value was 2.5, the maximum fitness error value of the IDE-BO-OBL algorithm was 3.32. Figure 9 (c) showed the boxplot of the IDE-BO-OBL algorithm in F7 at different p-values. In Figure 9 (c), when the p-value was 0.5, the minimum fitness error value of the IDE-BO-OBL algorithm was 4.12. When the p-value was 2.5, the maximum fitness error value of the IDE-BO-OBL algorithm was 5.15. Figure 9 (d) showed the box plot of the IDE-BO-OBL algorithm in F10 at different p-values. In Figure 9 (d), when the p-value was 0.5, the minimum fitness error value of the IDE-BO-OBL algorithm was 5.77. When the p-value was 2.5, the maximum fitness error value of the IDE-BO-OBL algorithm was 5.87. In summary, the IDE-BO-OBL algorithm had a small deviation in the calculation process and can be applied to complex functions.

III. B. Analysis of Application Results of IDE Indoor Layout with RLS

To verify the effectiveness of the IDE model with RLS in indoor layout design, simulation experiments were conducted on AutoCAD 2010 software. Table 1 shows the simulation experiment conditions.

Table 1: Experimental condition table

Environment	Argument	
Host machine	Intel(R)Core (TM) i3 RAM	
Operating system	Windows 10 flagship AutoCAD 2010	
Two-dimensional design software	MATLAB 2010	CPU 530@2.93GHz, 4.00GB
Programming language Problem dimension	30	
max_FEs	10000*D	

To test the application effect of the model in interior layout design, the satisfaction values of interior layout design users based on IDE-BO-OBL model, COBL model, and QOBL model were tested separately under European classical style and modern minimalist style. The experimental results are shown in Figure 10.

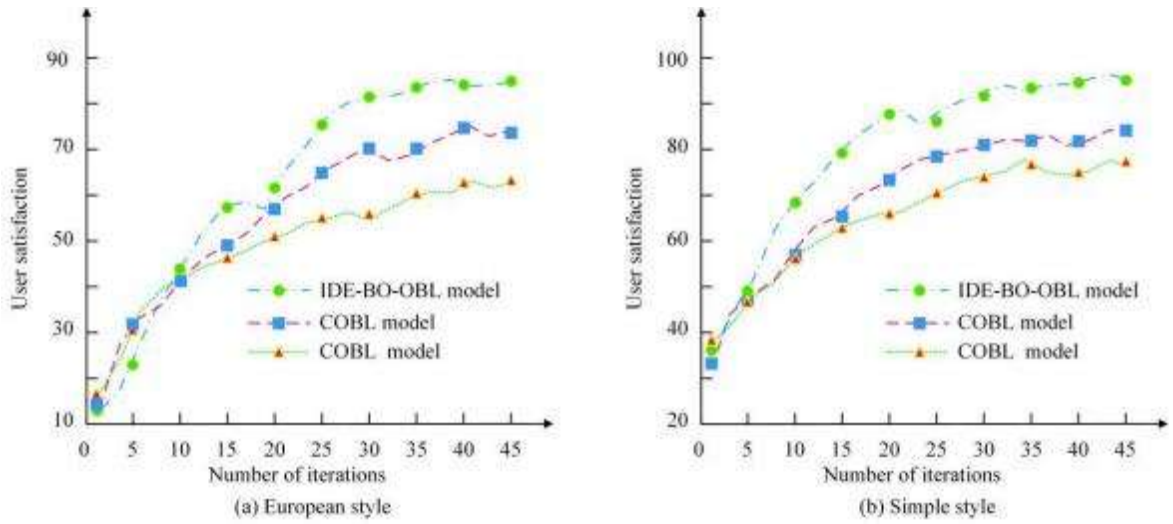

Figure 10: Statistical Results of User Satisfaction for Different Models

Figure 10 (a) showed the statistical results of user satisfaction with European style space design based on different models. In Figure 10 (a), for the European style spatial design, the IDE-BO-OBL model had a high level of user satisfaction, with the highest satisfaction value reaching 88 in the later stages of iteration. Although the user satisfaction of the COBL model increases with the number of iterations, it has the lowest user satisfaction among the three types of models. Figure 10 (b) showed the statistical results of user satisfaction with minimalist style space design based on different models. In Figure 10 (b), for the minimalist style space design, the IDE-BO-OBL model had a high level of user satisfaction, with the highest satisfaction value reaching 98 in the later stages of iteration. The COBL model had the lowest user satisfaction, and when the number of iterations was 45, the highest user satisfaction value was 78. To further demonstrate the superiority of the IDE-BO-OBL model, the Wilcoxon signed rank test was used to validate the performance of the IDE-BO-OBL model, COBL model, and

QOBL model. The results of the Wilcoxon test are presented in Table 2, which showed the statistical comparison between the two groups.

Table 2: Statistical comparison results of Wilcoxon tests

/	/	t∈ [1,10]	t∈ [11,30]	t∈ [31,40]	t∈ [1,40]
"European" style	IDE-BO-OBL model COBL model QOBL model IDE-BO-OBL model	0.871 1.000 0.873 0.470	0.332 0.093 0.001 0.003	0.003 0.003 0.003 0.165	0.256 0.110 0.018 0.166
"Simple" style	COBL model	0.471	0.001	0.004	0.007
	QOBL model	0.378	0.005	0.004	0.004

According to Table 2, in the European classical style, the p-values of the IDE-BO-OBL model in different four-time intervals were 0.871, 0.332, 0.003, and 0.256, respectively. In the modern minimalist style, the p-values of the IDE-BO-OBL model in four different time intervals were 0.470, 0.003, 0.165, and 0.166, respectively. In summary, the performance difference between the IDE-BO-OBL model and the other two models was not significant within the range of [1,10]. In the range of [11,30], the performance advantage of the IDE-BO-OBL model gradually became apparent, and in the range of [31,40], the performance advantage of the IDE-BO-OBL model was more obvious.

IV. Discussion And Conclusion

IV. A. Discussion

This study is based on IDE algorithm and RLS to optimize indoor layout design system. To test the research algorithm, the superiority of IDE-BO-OBL algorithm and the most advanced OBL variant algorithm in multiple aspects is studied. This study discusses algorithm efficiency, optimization effectiveness, applicability, and practical applications.

Experimental data showed that under different dimensional conditions, the IDE-BO-OBL algorithm had significantly better running time than COBL and QOBL algorithms in benchmark function testing, especially when dealing with complex combination function F10, where its running time advantage was more prominent. As the dimension increased, the running time of all algorithms increased. However, the growth rate of IDE-BO-OBL algorithm was relatively slow, reflecting its good scalability and stability. In terms of optimization performance, the IDE-BO-OBL algorithm could quickly find or approach the global optimal solution. Moreover, compared to COBL and QOBL algorithms, it had a lower risk of falling into local optima. In functions such as F1 and F5, the IDE-BO-OBL algorithm found the global optimum, while the COBL and QOBL algorithms exhibited varying degrees of local optima. This result was consistent with Li et al.'s research on a learning adaptive variable speed whale optimization algorithm based on quadratic opposition and chaos strategy. It proved the superiority of IDE-BO-OBL algorithm in optimization effect and provided a more accurate and reliable optimization solution for indoor layout design [22].

In practical applications, the IDE-BO-OBL algorithm showed good applicability in different types of benchmark functions and could find the global optimal solution in various complex scenarios, which made the IDE-BO-OBL algorithm widely applicable in the field of indoor layout design. Whether in European classical style or modern minimalist style, the IDE-BO-OBL model could improve user satisfaction and achieved good design results through iterative optimization. As the number of iterations increased, the performance advantage of the IDE-BO-OBL model gradually became apparent, especially in the later stages of iteration, where its user satisfaction value was significantly higher than other models, further proving its superiority in practical applications. The study applied the IDE-BO-OBL algorithm to the field of interior layout design and verified its effectiveness and feasibility in practical scenarios. The experimental results showed that the user satisfaction of the IDE-BO-OBL model was significantly higher than that of the COBL and QOBL models in both European classical style and modern minimalist style, and its advantages became more apparent with the increase of iteration times. This was consistent with the conclusion drawn by Biswas et al. in their study on improving the grey wolf optimizer's best and worst orthogonal inverse learning [23].

In summary, the IDE-BO-OBL algorithm provides strong support for the intelligence and efficiency of indoor layout design due to its efficient computational efficiency, superior optimization results, wide applicability, and good practical application effects.

IV. B. Conclusion

A new indoor layout design method based on IDE-BO-OBL algorithm was proposed to address the complexity of modern indoor space layout design and personalized user needs. Performance analysis was conducted on the IDE-BO-OBL algorithm, and experimental results showed that as the dimensionality increased, the running time of all three types of algorithms increased. The running time of the IDE-BO-OBL algorithm was more advantageous compared to the two advanced OBL variants. As evidenced by the convergence curves of the three algorithms on F1, F5, F7, and F10, the IDE-BO-OBL algorithm was capable of identifying the global optimal state across a range of benchmark functions. This suggested that the IDE-BO-OBL algorithm possesses robust global search capabilities and adaptability. The evaluation and analysis of the IDE-BO-OBL model showed that for European style spatial design, the user satisfaction of the IDE-BO-OBL model was relatively high, with the highest satisfaction value reaching 88 in the later stages of iteration. For minimalist style space design, the IDE-BO-OBL model had a high level of user satisfaction, with the highest satisfaction value reaching 98 in the later stages of iteration. It can be concluded that the effectiveness and practicality of the IDE-BO-OBL model in the field of interior layout design also strongly supports its application in more diversified and personalized interior design scenarios in the future. The limitation of this study is that the selection of experimental datasets and benchmark functions may not fully cover all practical design scenarios. It is recommended that future research expand the experimental scope, incorporate a greater variety of design cases, and include more intricate design requirements. This will facilitate a more comprehensive assessment of the performance and applicability of the IDE-BO-OBL algorithm.

Furthermore, the IDE-BO-OBL algorithm can be integrated with other sophisticated technologies, such as deep learning and augmented learning, to enhance its intelligence and user experience in interior space layout design.

References

[1] Alizadeh Z, Yazdi J. Calibration of hydrological models for ungauged catchments by automatic clustering using a differential evolution algorithm: The Gorganrood river basin case study. Journal of hydroinformatics, 2023, 25(4):645-662.

[2] Chen Z, Cao J, Zhao F. A Grouping Cooperative Differential Evolution Algorithm for Solving Partially Separable Complex Optimization Problems.Cognitive Computation, 2023, 15(7):956-975.

[3] Li W, Ye X, Huang Y. Adaptive Dimensional Learning with a Tolerance Framework for the Differential Evolution Algorithm. Complex System Modeling and Simulation, 2022, 2(1):59-77.

[4] Niu J, Liu Z, Pan Q. Conditional self-attention generative adversarial network with differential evolution algorithm for imbalanced data classification. Acta aeronautica sinica, 2023, 36(3):303-315.

[5] Kar M K, Kumar S, Singh A K. Reactive power management by using a modified differential evolution algorithm. Optimal Control Applications and Methods, 2023, 44(2):967-986.

[6] Li L X, Li J L, Nie J F. Parameter Adaptive Elite Mutation Differential Evolution Algorithm[J]. Journal of Chinese Computer Systems, 2023, 44(8):1693-1706.

[7] Khosla T, Verma O. An Efficient Particle Swarm Optimization with Fusion ofFigurate Opposition-based learning and Grey Wolf Optimizer. 2022 International Conference on Computing, Communication, and Intelligent Systems, 2022:208-213.

[8] Si T. 2D MRI registration using glowworm swarm optimization with partial opposition-based learning for brain tumor progression. Pattern analysis and applications: PAA, 2023, 26(3):1265-1290.

[9] Abdollahpour A, Rouhi A, Pira E. An improved gazelle optimization algorithm using dynamic opposition-based learning and chaotic mapping combination for solving optimization problems. The Journal of Supercomputing, 2024, 80(9):12813-12843.

[10] Yang X, Yu H, Fan G. DEJIT: A Differential Evolution Algorithm for Effort-Aware Just-in-Time Software Defect Prediction. International Journal of Software Engineering and Knowledge Engineering, 2021, 31(3):289-310.

[11] He M, Che J, Jiang W W B. A novel decomposition-denoising ANFIS model based on singular spectrum analysis and differential evolution algorithm for seasonal AQI forecasting. Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology, 2023, 44(2):2325-2349.

[12] Qinghua G, Yifan P, Song Q J. Multimodal multi-objective optimization based on local optimal neighborhood crowding distance differential evolution algorithm. Neural computing & applications, 2024, 36(1):461-481.

[13] Chu J, Li J, Zong S T. Parameter estimation of Wiener-Hammerstein system based on multi-population self-adaptive differential evolution algorithm. Engineering computations: International journal for computer-aided engineering and software, 2023, 40(9):2248-2269.

[14] Zhao J, Gan C M, Liu Z H. Differential Evolution Hemivariational Inequalities with Anti-periodic Conditions. Acta Mathematica Sinica,English Series, 2024, 40(4):1143-1160.

[15] Liao W, Xiao Y X, Jia X. A SpiderMonkey Optimization Algorithm Combining Opposition-Based Learning and Orthogonal Experimental Design. Computers, materials, and continuum, 2023, 76(9):3297-3323.

[16] Shuidong M, Yiming F, Le X L. Ameliorated grey wolf optimizer with the best and worst orthogonal opposition-based learning. Soft computing: A fusion of foundations, methodologies and applications, 2024, 28(4):2941-2965.

[17] Wang L, Li J, Yan X. A variable population size opposition-based learning for differential evolution algorithm and its applications on feature selection. Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies, 2024, 54(1):959-984.

[18] Chen L, Ma L, Li L. Enhancing sine cosine algorithm based on social learning and elite opposition-based learning. Computing, 2024, 106(5):1475-1517.

[19] Wang Z, Ding H, Yang Z H P. Advanced orthogonal opposition-based learning-driven dynamic salp swarm algorithm: Framework and case studies. IET control theory & applications, 2022, 16(10):945-971.

[20] Boroujeni S P H, Pashaei E. A hybrid chimp optimization algorithm and generalized normal distribution algorithm with opposition-based learning strategy for solving data clustering problems. Iran Journal of Computer Science, 2024, 7(1):65-101.
[21] Amin S N, Shivakumara P, Jun T X. An Augmented Reality-Based Approach for Designing Interactive Food Menu of Restaurant Using Android. Artificial Intelligence and Applications. 2023, 1(1): 26-34.
[22] Li M, Xu G, Lai Q. A chaotic strategy-based quadratic Opposition-Based Learning adaptive variable-speed whale optimization algorithm. Mathematics and Computers in Simulation (MATCOM), 2022, 193(6):71-99.
[23] Biswas S, Shaikh A, Ezugwu E S. Enhanced prairie dog optimization with Levy flight and dynamic opposition-based learning for global optimization and engineering design problems. Neural Computing and Applications, 2024, 36(19):11137-11170.