Computer Modeling in Engineering & Sciences

Tech Science Press

https://doi.org/10.32604/cmes.2026.084683

ARTICLE

Adaptive Evolution of Metaheuristic Update Strategies Using Genetic Programming for Remote Sensing Image Fusion[#]

Jeng-Shyang Pan[1,2,3], Wenda Li[2], Shu-Chuan Chu[4,*], Zhi-Gang Du[5], Hongmei Yang[2] and Lingping Kong[6]

[1]School of Information Engineering, Yango University, Fuzhou, China
[2]College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, China
[3]Department of Information Management, Chaoyang University of Technology, Taichung City, Taiwan
[4]School of Artificial Intelligence, Nanjing University of Information Science and Technology, Nanjing, China
[5]School of Transportation and Logistics, Southwest Jiaotong University, Chengdu, China
[6]Faculty of Electrical Engineering and Computer Science, VSB-Technical University of Ostrava, Ostrava, Czech Republic

*Corresponding Author: Shu-Chuan Chu. Email: scchu0803@gmail.com

[#]Presented at the International Conference on Machine Intelligence Theory and Applications 2026; 2026 Feb 24–28; Dunedin, New Zealand

Received: 27 April 2026; Accepted: 29 July 2026; Published: 28 August 2026

ABSTRACT: Formulating efficient updating techniques is essential for the efficacy of metaheuristic algorithms. Traditional approaches, however, depend significantly on manually developed formulas and empirical intuition, which frequently constrain their adaptability and scalability across various optimization tasks. This research introduces a Genetic Programming-based Metaheuristic framework, referred to as GP-MAs, designed to autonomously develop and enhance symbolic update rules for metaheuristic algorithms. Within the suggested GP-MAs architecture, genetic programming (GP) is integrated into the learning phase of the Growth Optimizer (GO) to dynamically formulate symbolic update equations, hence enhancing the algorithm's adaptability to diverse optimization landscapes. A hybrid algorithm, termed GPbasedGO, is constructed based on GO to validate the effectiveness of the proposed framework. Comprehensive experiments on the CEC2022 benchmark suite illustrate the competitive convergence characteristics, optimization efficacy, and competitive performance across benchmark problems of GPbasedGO in comparison to many prominent state-of-the-art techniques. The practical usability of the proposed approach is assessed in a multispectral and panchromatic image fusion challenge, where GPbasedGO attains competitive results across many image quality measures, including ERGAS, SAM, RMSE, UIQI, and CC. The findings demonstrate that the GP-MAs framework offers a versatile and semi-automated model for metaheuristic optimization, proficient in producing adaptive symbolic updating techniques for key optimization problems. This study presents a comprehensive validation of GP-assisted symbolic strategy evolution for metaheuristic optimization and suggests interesting avenues for further research in adaptive optimization.

KEYWORDS: Metaheuristic optimization; genetic programming; adaptive update strategy; image fusion

1 Introduction

The swift progression of artificial intelligence (AI) technology has resulted in its growing implementation across various sectors, including healthcare, transportation, and industrial manufacturing.

Notwithstanding this expansion, considerable hurdles persist, including technical impediments, data heterogeneity, and limitations dictated by particular application contexts. Comprehensive analyses of real-world problem models have revealed that these issues frequently display nonlinear, non-differentiable, and intricate properties. In this setting, meta-heuristic algorithms (MAs) have emerged as potent instruments for global optimization, providing a flexible and structure-agnostic approach to tackle these difficulties [1,2]. In contrast to conventional optimization methods, MAs are engineered to investigate intricate search spaces by emulating natural processes, biological evolution, or physical phenomena probabilistically. These algorithms utilize problem-specific knowledge to effectively pursue high-quality or near-optimal solutions. In recent years, meta-heuristic methodologies have exhibited exceptional efficacy across a diverse array of applications, encompassing wireless sensor networks (WSNs) [3,4], cloud computing resource allocation [5], communication security [6], intelligent manufacturing systems [7], machine learning [8], and numerous artificial intelligence endeavors [9,10].

The taxonomic classification of MAs is based on their foundational inspiration mechanisms, which are systematically divided into four principal categories as illustrated in Fig. 1. Metaheuristic algorithms are systematically categorized into four fundamental groups according to their sources of inspiration: Evolutionary-based algorithms, modeled on biological evolution principles, employ iterative population refinement through crossover, mutation, and selection mechanisms, with prominent implementations including Genetic Algorithm (GA) [11], Differential Evolution (DE) [12], Evolution Strategy (ES) [13], Biogeography-Based Optimization (BBO) [14], and Immunization Algorithm (IA) [15]; Swarm intelligence algorithms, derived from collective biological behaviors, leverage decentralized coordination strategies as exemplified by Particle Swarm Optimization (PSO) [16], Ant Colony Optimization (ACO) [17], Artificial Bee Colony (ABC) [18], Grey Wolf Optimization (GWO) [19], Whale Optimization Algorithm (WOA) [20], and Mountain Gazelle Optimization (MGO) [21]; Physics-based algorithms, grounded in physical/chemical laws, utilize force-directed solution space exploration techniques such as Simulated Annealing (SA) [22], Gravitational Search Algorithm (GSA) [23], Sine Cosine Algorithm (SCA) [24], and Multi-Verse Optimization (MVO) [25]; Human behavior-inspired algorithms, emulating cognitive-social dynamics, implement social learning paradigms including Teaching-Learning-Based Optimization (TLBO) [26], Harmony Search (HS) [27], Social Group Optimization (SGO) [28], and Special Forces Algorithm (SFA) [29]. This classification framework (Fig. 1) systematically organizes algorithmic design principles, wherein evolutionary methods formalize optimization through Mendelian genetics, swarm systems utilize stochastic agent-based models, physics-inspired techniques emulate natural forces, and human-centric approaches incorporate experiential adaptation mechanisms.

In recent years, research initiatives in MAs have focused on three primary trends. One of these paths entails the creation of wholly new algorithmic frameworks. The suggested Gannet Optimization Algorithm (GOA) [30], inspired by the foraging activities of gannets, employs U-shaped and V-shaped diving patterns for exploration and utilizes abrupt turns combined with random walks to enhance solutions. In a similar vein, Wang et al. [31] presented the Status-based Optimization (SBO) method, which emulates the human aspiration for status elevation. A different research direction focuses on the hybridization of various computer methods to collaboratively combine their respective benefits. Liang et al. [32] exemplified this approach by developing the hybrid IMPAEO algorithm, which integrates an enhanced marine predators algorithm with equilibrium optimization. Meanwhile, Song et al. [33] devised the HADEFP methodology through the strategic hybridization of differential evolution and flower pollination algorithms. A third research direction focuses on improving algorithmic efficacy by incorporating creative methodologies or parameter tweaking to boost performance across various problem domains [34].

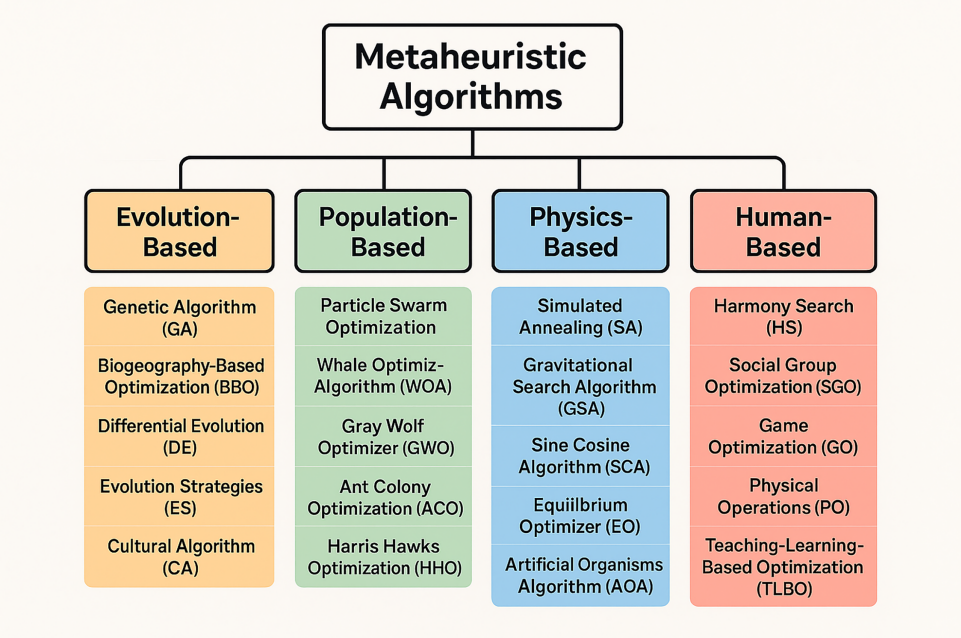


Figure 1: A brief classification of metaheuristic algorithms.

Notwithstanding the formal diversity of metaheuristic algorithms, their fundamental evolutionary framework remains predominantly dependent on manually crafted update rules. The iterative formulations of algorithms, such as speed updates for Particle Swarm Optimization and position updates for Grey Wolf Optimization, are typically developed manually by researchers via empirical induction or phenomenological analogies. This technique demands a profound comprehension of the target problem's characteristics and involves the adjustment of parameter weights following extensive trial-and-error experimentation. The constraints of these manual designs are becoming increasingly evident:

- **Empirical reliance and generalization deficiencies:** Manual formulations are typically tailored for particular issue circumstances, and their sensitivity to parameters and inflexible strategies result in considerable performance variability when utilized across different domains. For instance, PSO's fixed-rate update model often succumbs to local optimization in high-dimensional multi-peak problems, whereas GWO's linear convergence mechanism finds it challenging to reconcile the demands of dynamic exploration and exploitation.
- **Absence of automation capability:** The update protocols of current algorithms lack adaptive adjustment capabilities and are unable to dynamically generate or alter strategies during the optimization process. While hybrid algorithms can mitigate this problem, their foundational layers remain dependent on predetermined static rules, complicating the adaptation to abrupt modal shifts in intricate optimization landscapes.
- **Excessive utilization of computing resources:** To enhance algorithmic robustness, researchers must regularly recalibrate formula parameters or devise novel hybrid solutions from the ground up. This requires considerable professional work and computational resources, hence limiting large-scale engineering applications.

To overcome these constraints, Meta-Black-Box Optimization (MetaBBO) has recently emerged as a potential approach, utilizing meta-learning to automate the design and adaptation of BBO algorithms with

minimum expert involvement [35,36]. MetaBBO generally employs a bi-level framework, wherein a meta-level policy monitors the optimization dynamics of a low-level BBO process and adaptively formulates algorithmic designs based on performance feedback over a problem distribution. Current MetaBBO research may be classified as algorithm selection [37], algorithm configuration [37], solution manipulation [38], and algorithm generation [39]. Diverse learning paradigms, encompassing reinforcement learning [40], supervised learning [41], neuroevolution [42], and LLM-based in-context learning [43], have been investigated across various optimization contexts, including single-objective, multi-objective, multi-modal, large-scale, and multi-task optimization [44].

Notwithstanding the swift advancement of MetaBBO [45], the majority of current methodologies primarily depend on neural-network-based meta-policies—especially reinforcement learning [46]—to facilitate adaptive algorithm design via online state-action decision-making. While policy-learning-based frameworks exhibit robust performance, they generally necessitate meticulously crafted state representations, intricate reward functions, and substantial training interactions. This elevates algorithmic complexity and drastically diminishes interpretability. Furthermore, dependence on online policy inference incurs significant extra computing burden throughout the optimization process.

This article introduces a Genetic Programming-based Metaheuristic framework, referred to as GP-MAs, which emphasizes the evolving explicit symbolic update rules using genetic programming in an offline manner at the algorithmic level, in contrast to policy-learning-based MetaBBO frameworks. Genetic Programming (GP), a hyper-heuristic algorithm that autonomously generates computational programs [47], offers a potential foundation for tackling these challenges [48]. Rather of acquiring an implicit control policy, GP-MAs utilize the symbolic regression ability of GP to develop update formulas in an offline manner. The proposed system substitutes manually developed update rules with evolving symbolic expressions, therefore diminishing dependence on expert knowledge, preserving adaptability across various optimization settings, and reducing the online decision-making burden compared with neural MetaBBO approaches characteristic of neural MetaBBO models. In contrast to reinforcement-learning-based MetaBBO systems that primarily enhance implicit decision policies via online interaction, the proposed GP-MAs framework emphasizes the offline symbolic development of explicit update rules or search operators within metaheuristic algorithms. GP-MAs develop symbolic update rules utilizing predefined terminal sets (core MA variables) and function sets (mathematical/logical operators), assessing candidate formulations via a specified objective function to deliver adaptive update strategies for representative metaheuristic frameworks.

To implement and evaluate the efficacy of the proposed GP-MAs architecture, the Growth Optimizer (GO) [49], an algorithm derived from human learning and reflective behaviors, is utilized as a baseline human-inspired optimizer. This study selects GO as a sample human-inspired optimizer to assess the viability of the proposed GP-MAs system. This study enhances the GP-MAs framework by reconfiguring the learning-stage updating process of GO to create a hybrid variation called GPbasedGO. This integration incorporates GP-based symbolic strategy development into the learning phase of GO, establishing an adaptive symbolic updating mechanism. GPbasedGO produces a variety of candidate strategies using syntax tree encoding and employs a selection strategy considering convergence and exploration performance to equilibrate convergence velocity and exploratory capacity. To assess the efficacy of the proposed strategy, systematic tests are performed on the CEC2022 benchmark suite and a multispectral-panchromatic image fusion job. The benchmark tests assess convergence behavior and optimization performance, while the image fusion problem serves as an application case study of the proposed framework in typical remote sensing optimization scenarios.

The subsequent sections of this work are organized as follows. Section 2 thoroughly explains the fundamental approaches, including the Growth Optimizer (GO) algorithm, Genetic Programming (GP),

and panchromatic-multispectral picture fusion techniques. Section 3 outlines the GP-MAs framework in conjunction with the architectural design and execution of the GPbasedGO algorithm. Section 4 delineates empirical findings and comparative studies. Ultimately, Section 5 consolidates the research findings and delineates potential avenues for future investigation.

2 Related work

This chapter will carefully outline the theoretical underpinnings and technical procedures relevant to this research, focusing on Growth Optimizer (GO), Genetic Programming (GP), and the key approaches to multispectral and panchromatic picture fusion. This material provides theoretical support for the following technique and experimental analysis.

2.1 Growth Optimizer

The GO algorithm, as a fledgling intelligent optimization approach, emulates the learning mechanism of reflection and self-improvement that is inherent in human development, with broad global search capabilities and considerable versatility. This section explains the core structure of the GO algorithm, including the initialization strategy, learning process, and the crucial notion of the reflection phase, which serves as a foundation for future algorithm modifications.

2.1.1 Initialization

The initialization process commences with the individuals of the $0th$ generation. For the ith individual $X_i(0)$ in this generation, its jth dimension component $x_{i,j}(0)$ resides inside the interval $[x_{i,j}^{\min}, x_{i,j}^{\max}]$. Here, $x_{i,j}^{\min}$ denotes the lower limit and $x_{i,j}^{\max}$ signifies the upper bound for the jth dimension. The variable i denotes the ith individual in the population, while j signifies the jth dimension. The population size is represented by N_p, while D signifies the number of dimensions. The population initialization is executed via a uniform distribution method, as illustrated in Eq. (1).

$$x_{i,j}(0) = x_{i,j}^{\min} + \text{rand}(0,1) \cdot (x_{i,j}^{\max} - x_{i,j}^{\min}) \tag{1}$$

Eq. (1) demonstrates the initialization of each individual's components within defined limits by a uniform random distribution, hence guaranteeing a varied beginning population across all dimensions and individuals.

2.1.2 Learning Phase

During the learning phase of the GO algorithm, individuals analyze their differences, explore the reasons behind these gaps, and learn from them, resulting in significant growth. To facilitate this process, four key types of gaps are defined: the difference between the leader and an elite individual (Gap_1), between the leader and a lower-ranked individual (Gap_2), between an elite and a lower-ranked individual (Gap_3), and between two randomly selected individuals (Gap_4). The mathematical model for these gaps is as follows:

$$\begin{cases} \text{Gap}_1 = \vec{x}_{\text{best}} - \vec{x}_{\text{better}} \\ \text{Gap}_2 = \vec{x}_{\text{best}} - \vec{x}_{\text{worse}} \\ \text{Gap}_3 = \vec{x}_{\text{better}} - \vec{x}_{\text{worse}} \\ \text{Gap}_4 = \vec{x}_{\text{L1}} - \vec{x}_{\text{L2}} \end{cases} \tag{2}$$

In this model, $\vec{x}_{\text{best}}$ denotes the leader of the group, while $\vec{x}_{\text{better}}$ represents an elite individual. Together, they form the upper tier of the social hierarchy. $\vec{x}_{\text{worse}}$ indicates a lower-ranked individual within the population. $\vec{x}_{\text{L1}}$ and $\vec{x}_{\text{L2}}$ are randomly chosen individuals, distinct from the i-th individual. Gap_k (where $k = 1, 2, 3, 4$) signifies the difference between two individuals, offering learners valuable insights into these discrepancies for their benefit. It is important to note that within the current iteration (it) of the GO algorithm, to streamline the update and selection of individuals, GR should be sorted in ascending order.

The Learning Factor (LF_k) is used to measure the strength of an individual's learning from the kth set of gaps and is calculated as:

$$LF_k = \frac{\|\text{Gap}_k\|}{\sum_{k=1}^{4} \|\text{Gap}_k\|}, \quad (k = 1, 2, 3, 4) \tag{3}$$

where LF_k reflects the relative size of the Euclidean distance of the group k gap among all gaps. The larger the LF_k, the greater the impact of the gap on the individual's learning, and the more the individual learns from that gap. For example, if Gap_1 has a larger LF_1, it indicates that the gap between leaders and elites is larger, and the individual needs to learn more from that gap to close the gap with leaders and elites.

The self-perception factor (SF) is used to assess an individual's own situation and is calculated using the formula:

$$SF = \frac{GR}{GR_{\max}} \tag{4}$$

where GR represents the individual's growth resistance (with smaller values being better), and $GR_{\max}$ denotes the maximum growth resistance observed across all individuals. The scaling factor SF has values within the range $[0, 1]$. The smaller the value of SF, the smaller the individual's GR, indicating a higher position in the social hierarchy. Such individuals are more likely to engage in localized search (refined exploration mode). Conversely, the larger the value of SF, the greater the individual's GR, placing them in a lower social tier. These individuals need to conduct more global searches (extensive exploration mode).

Knowledge Acquisition (KA_k) is the core process in the learning phase, where individuals acquire knowledge from gaps through KA_k, which is calculated as:

$$\vec{K}A_k = SF \cdot LF_k \cdot \vec{Gap}_k, \quad (k = 1, 2, 3, 4) \tag{5}$$

where KA_k is the amount of knowledge an individual acquires from the kth set of gaps. SF and LF_k together influence the size of KA_k. For example, an individual in a higher social stratum (small SF) who learns more information (LF_k) from a given gap will acquire a correspondingly larger amount of knowledge.

Individuals need to update their position after acquiring knowledge in the learning phase with the update formula:

$$\vec{x}_i^{\text{It+1}} = \vec{x}_i^{\text{It}} + \vec{K}A_1 + \vec{K}A_2 + \vec{K}A_3 + \vec{K}A_4 \tag{6}$$

where x_i^{It} is the position of individual i at the current iteration It, and x_i^{It+1} is the position of individual i at the next iteration $It + 1$. Individuals grow by updating their position by absorbing the knowledge gained from the gap.

After updating, it needs to be verified whether the individual is actually progressing or not. If progress is made (i.e., the fitness value is lower), the updated individual is retained; if no progress is made, the new individual is retained with probability $P2$. This verification process ensures that individuals can truly improve

themselves after learning and avoids the algorithm from falling into local optimality. The specific formula is shown in Eq. (7).

$$\vec{x}_i^{\mathrm{it+1}} = \begin{cases} \vec{x}_i^{\mathrm{it+1}} & \text{if } f\left(\vec{x}_i^{\mathrm{it+1}}\right) < f\left(\vec{x}_i^{\mathrm{it}}\right) \\ \begin{cases} \vec{x}_i^{\mathrm{it+1}} & \text{if } r_1 < P_2 \\ \vec{x}_i^{\mathrm{it}} & \text{else} \end{cases} & \text{else} \end{cases} \tag{7}$$

where $r_1 \in [0,1]$ is a uniformly distributed random number, and $\mathrm{ind}(i)$ denote the ranking of the ith individual based on an ascending order of the GR value. In cases where the individual fails to update, the retention of newly acquired knowledge is governed by the parameter P_2, which is set to 0.001. Due to space limitations in Eq. (7), the full conditional statement for knowledge acquisition is expressed as: $r_1 < P_2$ and $\mathrm{ind}(i) \neq \mathrm{ind}(1)$. This condition ensures that, even if an individual update fails, there is a small probability (0.001) that the learned information will be retained and the individual will be carried over to the next generation. Moreover, it guarantees that the current global best individual remains unchanged, thereby avoiding potential disruptions in the convergence of the algorithm.

Through the above process, the learning phase of the GO algorithm not only helps individuals to acquire knowledge from different gaps, but also adaptively adjusts the direction and intensity of learning through the self-awareness factor and the learning factor, so as to realize the balance between global search and local development.

2.1.3 Reflection Phase

The mathematical model of the reflection phase is described by Eqs. (8) and (9) as follows:

$$\vec{x}_{i,j}^{\mathrm{it+1}} = \begin{cases} lb + r_4 \times (ub - lb) & \text{if } r_3 < AF \\ \vec{x}_{i,j}^{\mathrm{it}} + r_5 \times \left(R_j - \vec{x}_{i,j}^{\mathrm{it}}\right) & \text{if } r_2 < P_3 \\ \vec{x}_{i,j}^{\mathrm{it}} & \text{else} \end{cases} \tag{8}$$

$$AF = 0.01 + 0.99 \times \left(1 - \frac{FEs}{MaxFEs}\right) \tag{9}$$

The reflection phase processes each dimension of the individual with three possible operations:

- **Keep the original value:** If the random number $r_2 \geq P_3$, the individual's jth dimension remains unchanged. This means that the individual has not learned anything new from other individuals in that dimension, or that the current state is considered good enough and does not need to be changed.
- **Updated under the guidance of a senior individual:** If $r_2 < P_3$, the individual's j dimension is updated under the guidance of a senior individual, such as a leader or elite. The senior individual provides a goal direction toward which the individual moves, which helps the individual move closer to a more optimal solution in that dimension.
- **Reinitialization based on advanced individuals:** If $r_3 < AF$, the jth dimension of the individual is reinitialized with a small probability. The process of reinitialization is to randomly select a new value in the search space, which increases the diversity of the population, prevents the algorithm from converging prematurely, and helps to jump out of local optima.

In Eq. (9), FEs is the current number of evaluations and $MaxFEs$ is the maximum number of evaluations. As the algorithm iterates, AF gradually decays linearly from close to 1 to 0.01. At the beginning of the algorithm, AF is larger, and individuals have a higher probability of re-initialization, which helps

in the global search. As the iteration progresses, AF gradually decreases, the probability of individual re-initialization decreases, and the algorithm gradually shifts to local search to refine the current solution.

The reflection phase helps people recognize and correct problems by motivating them to analyze and develop themselves. This stage encourages individual self-improvement so that the population as a whole can develop toward a better solution. Individuals can improve themselves by taking guidance from more experienced people.

GO was selected as a sample human-inspired optimizer due to its structured learning mechanism and strong performance in complex search contexts, making it a good baseline for evaluating the suggested methods. Although GO is utilized as the baseline optimizer in this study, the proposed framework is not limited to it and could be expanded to incorporate other population-based metaheuristics.

2.2 Genetic Programming

Genetic programming is a self-contained programming technique based on evolutionary ideas that creates program structures adapted to individual tasks by mimicking natural selection and genetic processes. In this study, genetic programming is used to produce key operators in optimization models. This section will present the individual generation method to genetic programming and its key evolutionary processes, laying the theoretical groundwork for the integration of GO algorithms with GP.

2.2.1 Individual Initialization

Individuals in genetic programming are usually encoded in a tree structure, where non-leaf nodes represent functions and leaf nodes are terminal symbols. A function can be found in the set of functions $F = \{f_1, f_2, \ldots, f_n\}$, where f_i includes operators $(+, -, *, /$, Boolean operators, etc.) or functions $(\sin, \tan, \exp,$ etc.) as well as some expressions (cyclic expressions or conditional expressions, etc.). Terminators, on the other hand, can be selected from the set of terminators $T = \{t_1, t_2, \ldots, t_m\}$, which can be variables or constants.

In the first stage of the algorithm, an initial population with diversity and a certain complexity must be generated, a process known as individual initialization. To generate an initial population with diverse structure, GP frequently uses the following initialization methods:

Full Method: This method uses function nodes to completely fill the non-leaf positions under the set maximum depth limit, and terminal nodes only appear in the leaf positions, generating an individual tree with regular structure and consistent depth.

Grow Method: This method randomly selects function or terminal nodes to fill in the node positions at each level, which makes the individual structure more irregular in depth and branching and is suitable for exploring the diversity of the structure space.

Ramped Half-and-Half Method: Combining the above two methods, alternating between the full-full method and the growth method to generate individuals in different depth ranges is the most commonly used initialization strategy, which is capable of balancing the structural diversity of the population with the search breadth.

2.2.2 Basic Operations of Genetic Programming

The main evolutionary process of genetic programming is made up of three basic operations: selection, crossover, and mutation, which mimic the evolutionary process of organisms and drive the ongoing optimization of programmed people under the guidance of the fitness assessment mechanism.

(1) The selection operation selects which people have the opportunity to enter the next generation and reflects the supremacy of genetic programming. Its primary purpose is to identify highly adaptable programmed individuals based on their particular fitness levels. Commonly used selection strategies are:

Roulette Wheel Selection: The probability of an individual being selected is directly proportional to its fitness value, and is suitable for use in situations where there is a large difference in fitness.

Tournament Selection: A number of individuals are randomly selected from the population for a tournament to select the one with the highest fitness, which has a strong selection pressure.

Rank Selection: Assigning probabilities according to the order of fitness to avoid monopolization of the breeding process by over-adapted individuals.

Adjusting the selection strategy appropriately can avoid the rapid loss of population diversity while maintaining the inheritance of superior genes.

(2) Crossover operation: Crossover is the main ?recombination? mechanism in genetic programming, which aims to combine the structural information of two good individuals to create potential high-quality offspring. As illustrated in Fig. 2. The basic process is as follows:

- Randomly select a subtree (substructure) from each of the two parents.
- Swap these two subtrees.
- Generating two new individuals as offspring.

(3) Mutation operation: The mutation operation introduces a small structural disturbance to prevent the algorithm from falling into the local optimum and maintain the diversity of the population. As illustrated in Fig. 3, in genetic programming, the most commonly used is subtree mutation:

- Randomly select a subtree in the individual as the mutation point.
- Replace the position with a newly generated random subtree.
- Form a new program structure.

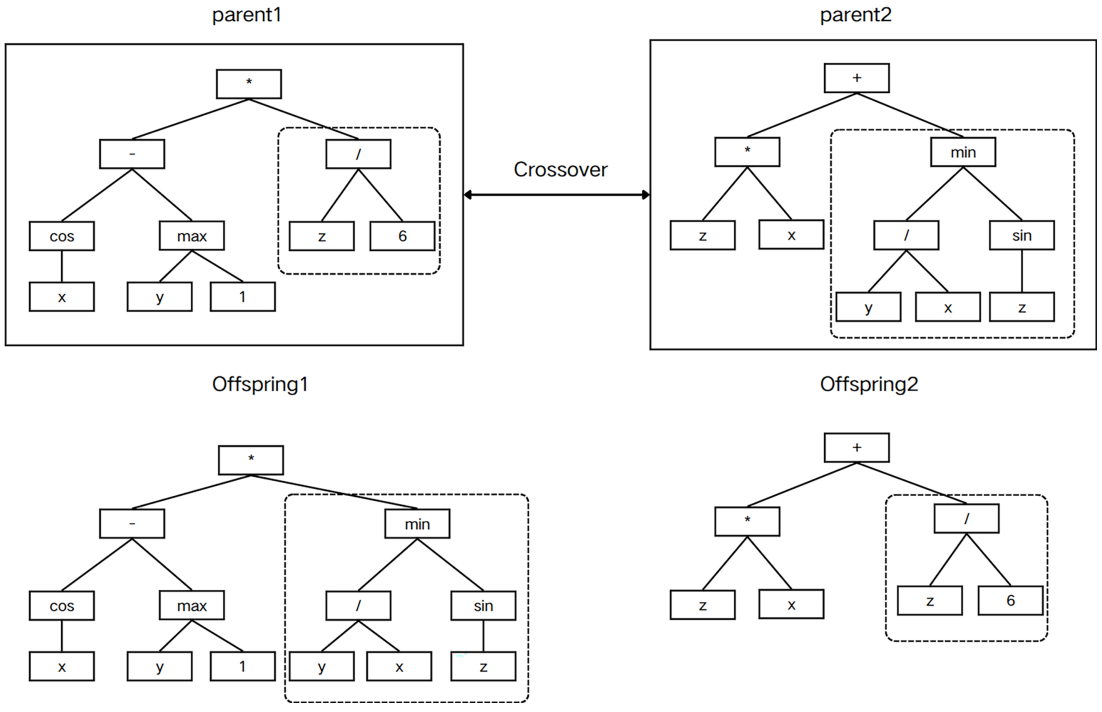


Figure 2: Sample diagram of crossover operation.

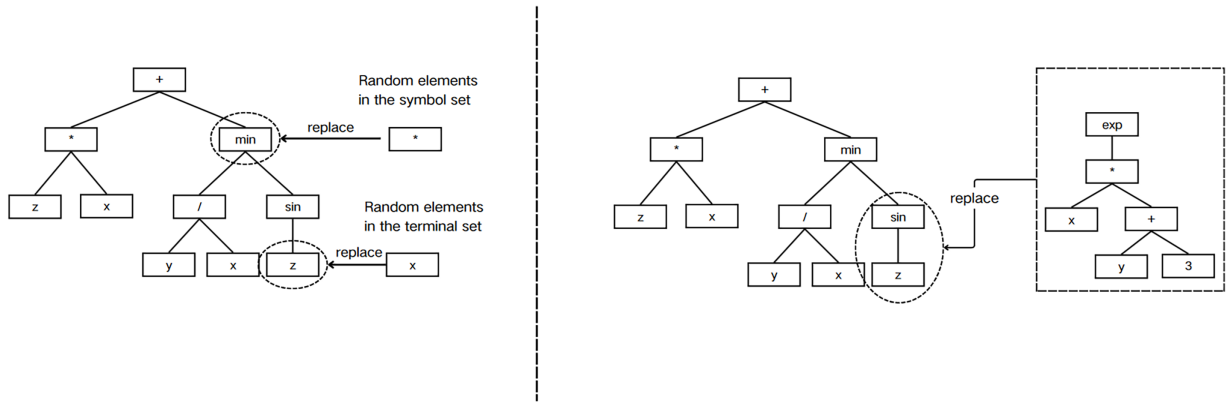


Figure 3: Sample diagram of mutation operation

In addition, there is also point mutation, which is to randomly replace a node in the individual (such as replacing a function symbol or variable symbol), which is less destructive but introduces less information. The mutation probability is usually set small to avoid destroying the existing excellent structure.

2.3 Panchromatic and Multi-Spectral Image Fusion

In the realm of remote sensing image processing, multi-source data fusion is an important technique for increasing image quality. Among these techniques, the fusion of panchromatic (Pan) and multispectral (MS) pictures is especially important. Fig. 4 shows a schematic diagram of the pan-sharpening process. By combining multispectral images with high spectral but low spatial resolution and panchromatic images with low spectral but high spatial resolution, the fusion process creates an image with both high spectral and spatial resolution. Remote sensing image fusion technology encompasses a variety of established methods, typically categorized as follows:

1. **Color Transformation Methods:** The Intensity-Hue-Saturation (IHS) transformation represents a widely adopted technique. By converting RGB color space into IHS space, this method replaces the intensity component with the high-resolution panchromatic image, thereby enhancing spatial resolution while preserving spectral content.
2. **Statistical Analysis Methods:** Principal Component Analysis (PCA) is a representative statistical method that decorrelates spectral bands into orthogonal principal components. The first principal component, which captures the largest variance, is typically replaced by the panchromatic image. While effective in maintaining spectral characteristics, PCA may fall short in capturing spatial details.
3. **Numerical Methods:** The Brovey transformation exemplifies a numerical approach, leveraging arithmetic operations within the RGB space to inject spatial information from the panchromatic image. It modulates the brightness component of the multispectral image while retaining its spectral fidelity.
4. **Hybrid Methods:** Hybrid fusion techniques integrate multiple fusion paradigms to leverage their respective strengths. The proposed fusion strategy in this study is a representative hybrid approach, integrating intelligent optimization algorithms with conventional fusion techniques to enhance overall performance. In this study, image fusion is used as an optimization-driven application scenario to evaluate the effectiveness of the proposed metaheuristic learning framework rather than proposing a new fusion algorithm.

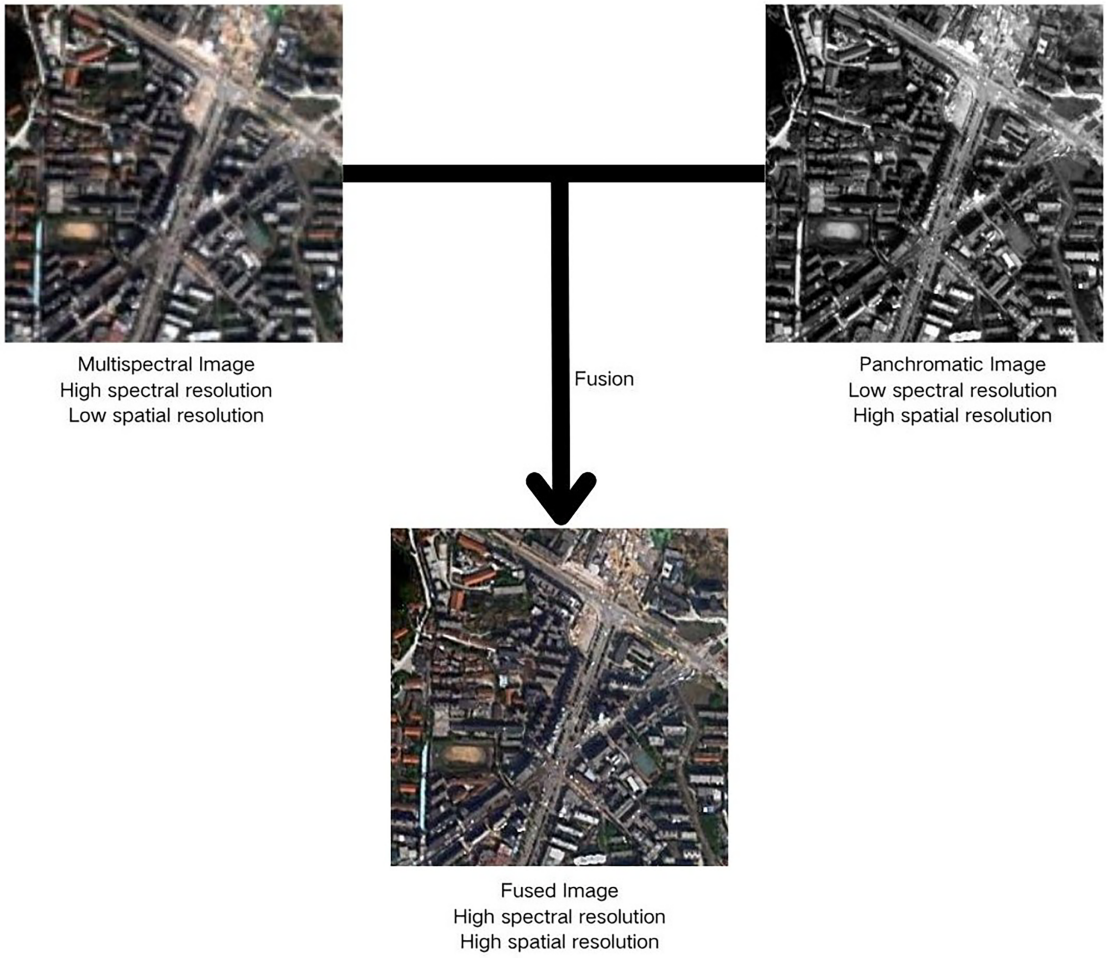


Figure 4: Schematic diagram of panchromatic image and multispectral image fusion.

Image fusion methods based on parameter optimization have emerged as a significant research focus. The core idea lies in constructing a mathematical mapping between the panchromatic and multispectral images. The general model is described as:

$$F = M + g(P - D) \tag{10}$$

where F is the fused high-resolution image, M denotes the multispectral band, P is the panchromatic image, D represents the detail component, and g is the mixing parameter that controls the balance of spatial and spectral contributions.

In the IHS framework, the intensity component I is formulated as:

$$I = \sum_{i=1}^{n} \alpha_i M_i \tag{11}$$

Here, α_i is a non-negative coefficient. In RGB images, α_i is commonly set to 1/3 to ensure equal contribution from each channel. For multispectral images with n bands, α_i is generally set to $1/n$ to maintain balanced spectral contribution across all channels.

To further reduce spectral distortion, the goal is to ensure that the panchromatic image closely approximates a linear combination of the multispectral bands:

$$P \approx \sum_{i=1}^{n} \alpha_i M_i \tag{12}$$

This leads to an optimization problem that minimizes the spectral discrepancy:

$$|P - \sum_{i=1}^{n} \alpha_i M_i|^2 \tag{13}$$

The optimal α_i can be determined via the Least Squares Method (LSM):

$$\alpha = (M^T M)^{-1} M^T P \tag{14}$$

This optimization framework effectively enhances the spatial resolution of multispectral images while preserving their spectral integrity, thereby supporting high-quality remote sensing applications.

In image processing, quantitative evaluation of fusion performance is crucial for verifying algorithm efficacy. Multiple evaluation metrics are commonly employed to assess spatial enhancement, spectral preservation, and visual fidelity.

1. Root Mean Square Error (RMSE): RMSE is a fundamental metric that quantifies pixel-level deviations between the fused image and the reference. It is defined as:

$$RMSE = \sqrt{\frac{1}{L \times W} \sum_{i=1}^{L} \sum_{j=1}^{W} (F_{i,j} - M_{i,j})^2} \tag{15}$$

where L and W denote the image dimensions, F and M represent pixel intensities of the fused and reference images, respectively. Lower RMSE indicates better fidelity to the reference image.

2. Relative Dimensionless Global Error in Synthesis (ERGAS): ERGAS evaluates the relative discrepancy between fused and reference multispectral images, accounting for resolution differences:

$$ERGAS = 100 \times \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left(\frac{RMSE_i}{\mu_i} \right)^2} \tag{16}$$

where n is the number of bands and μ_i is the mean of the ith band. Lower ERGAS implies higher overall image quality.

3. Spectral Angle Mapper (SAM): SAM measures spectral similarity using vector angles:

$$SAM = \arccos \left(\frac{\sum_i M_i \cdot F_i}{\sqrt{\sum_i M_i^2} \cdot \sqrt{\sum_i F_i^2}} \right) \tag{17}$$

Smaller SAM values reflect better spectral preservation.

4. Relative Average Spectral Error (RASE): RASE quantifies the mean spectral deviation relative to each band:

$$RASE = \frac{100}{\mu_M} \sqrt{\frac{1}{n} \sum_{i=1}^{n} RMSE_i^2} \tag{18}$$

Lower $RASE$ values indicate superior spectral consistency.

5. Universal Image Quality Index (UIQI): UIQI integrates brightness, contrast, and structural similarity:

$$UIQI = \frac{4\sigma_{MF}}{(\sigma_M^2 + \sigma_F^2)} \cdot \frac{2\mu_M\mu_F}{(\mu_M^2 + \mu_F^2)} \tag{19}$$

$UIQI$ values range from −1 to 1, with values closer to 1 indicating better overall quality.

6. Correlation Coefficient (CC): CC assesses linear correlation between the fused and reference images:

$$CC = \frac{\sum_{i,j}(M_{i,j} - \mu_M)(F_{i,j} - \mu_F)}{\sqrt{\sum_{i,j}(M_{i,j} - \mu_M)^2 \sum_{i,j}(F_{i,j} - \mu_F)^2}} \tag{20}$$

CC values close to 1 suggest strong similarity and effective information preservation.

Collectively, these evaluation metrics provide a robust foundation for comprehensive performance assessment in remote sensing image fusion, facilitating the development of more accurate and reliable fusion methodologies.

3 GP-MAs Framework and GO Algorithm Based on the GP-MAs Framework

This chapter provides a detailed introduction to the GP-MAs framework. It begins by outlining the framework's overall architecture and then goes into detail into the design of the relevant parameters. It also investigates the integration of Genetic Programming-based Metaheuristic Algorithms (GP-MAs) into the Growth Optimizer (GO) and gives a thorough examination of the updating mechanisms specifically designed for GO.

3.1 Overall Structure

Fig. 5 shows the architecture of the proposed GP-MAs-GO system, with red lines indicating data flow. This study's basic algorithm is the original Growth Optimizer (GO). The GP-MAs module is only included in the learning phase of GO, with the goal of constantly improving the learning method. The reflection phase and overall structural integrity of GO remain unaffected. In Step 1, researchers manually define the function set, terminal set, and objective function for evaluating the established learning algorithms. These capabilities are subsequently incorporated into the GP-MA system. Within the GP-MAs framework, a population of genetic programming (GP) individuals is established, with each representing a potential learning formula. Advantageous learning mechanisms are preserved by evolutionary processes like as selection, crossover, and mutation, while less successful ones are gradually destroyed.

The assessment process for each individual in the GP population comprises putting the candidate formula into the learning phase of GO and replacing the original update rule. The improved GO method is then applied to a set of benchmark functions, with the ultimate optimization performance serving as the fitness value for each individual. This ensures that only learning strategies that improve GO are kept.

In Step 2, the original handcrafted formula in GO is replaced with the most effective evolved formula obtained using the GP-MAs approach. The resulting algorithm, GPbasedGO, distinguishes itself from classic GO by using a dynamically changeable learning mechanism. In contrast to the fixed, human-engineered updating rules used in traditional GO, GPbasedGO adapts its learning method to changing issue landscapes and evolutionary stages.

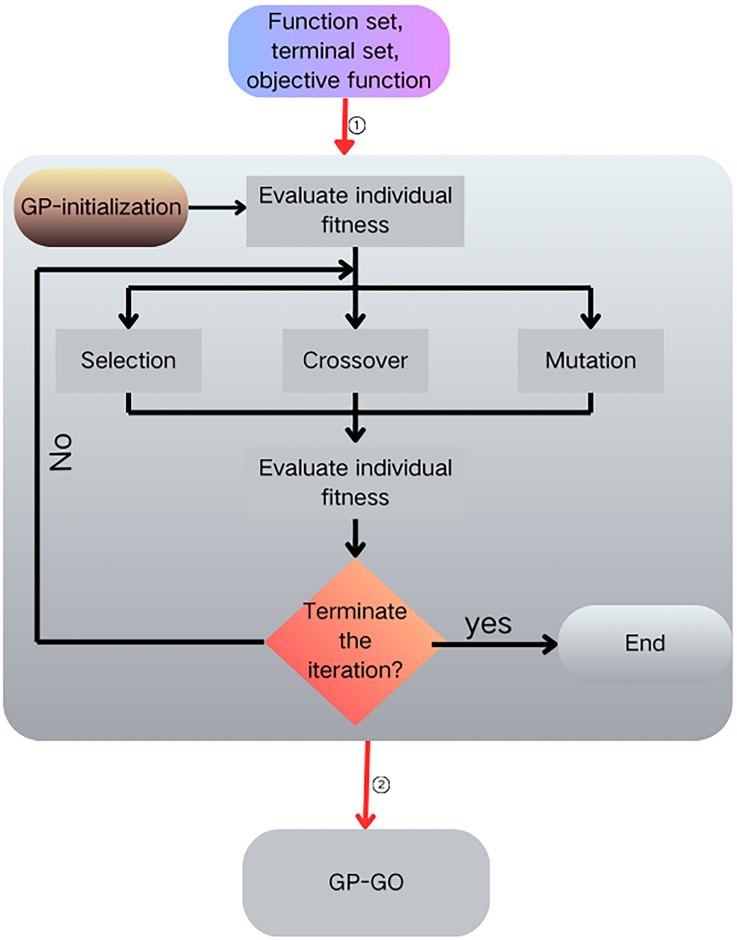


Figure 5: System framework diagram.

The growth of learning approaches via GP-MAs takes place offline, allowing the best formula to be used across several optimization issues without retraining. In practical optimization research, the manual creation of evolutionary formulas typically requires iterative parameter adjustments, variable alignments, heuristic experimentation, and comprehensive performance evaluations, resulting in significant instability and difficult cost assessment. The proposed GP-MAs framework replaces human design patterns with automated symbolic evolution, with the GP process performing variable matching and performance assessment on a continuous basis. While the offline GP evolution phase may require additional computational time, the created symbolic methodologies can be reused in future optimization projects, reducing redundant manual redesign efforts and extended human participation. Thus, the proposed framework provides a more adaptable and automated approach for symbolic strategy building in metaheuristic optimization.

Unlike traditional GP-based hyper-heuristics, which evolve entire algorithms or heuristics, GP-MAs evolve symbolic update rules nested within a fixed metaheuristic backbone. This constraint minimizes search space complexity while preserving the interpretability of the generated operators. In contrast to MetaBBO approaches, which train policy networks or selection strategies, GP-MAs directly optimize explicit mathematical update expressions.

3.2 The GP-MAs Framework

The GP-MAs (Genetic Programming-based Metaheuristic Algorithms) framework employs genetic programming to automatically generate update techniques for metaheuristic optimization algorithms. The system encodes search behaviors in adaptive mathematical expressions, which are then iteratively refined to increase optimization performance. Table 1 provides an overview of GP-MA parameter configurations. The GP parameter settings used in this study were based on commonly acknowledged configurations in the GP literature and preliminary empirical findings. These parameter sets intended to maintain an equilibrium between evolutionary diversity and computational stability as symbolic strategies evolved. Nonetheless, the efficiency of GP-based symbolic evolution may be influenced by hyperparameter selection. A more extensive sensitivity analysis of Gaussian Process hyperparameters will be addressed in future studies.

Table 1: Parameter settings of the GP-EAs framework.

Parameter	Value
Population size (PS)	100
Initialization method	Half depth-first, half breadth-first
Maximum iterations (MAXIT)	200
Tree depth (MAXD/MIND)	9/7
Elitism rate	20%
Crossover/Mutation probability	0.8/0.2

The detailed steps of the GP-MAs framework are as follows:

1. **Initialization:** A population of individuals is randomly initialized using a predefined set of functions and terminals. To enhance structural diversity, half of the individuals are generated using a depth-first growth method, while the remaining half use breadth-first construction.
2. **Expression Construction and Evaluation:** Each individual encodes a candidate metaheuristic component (e.g., an update rule or control policy) as a tree structure. These individuals are converted into executable expressions via in-order traversal and then embedded into a target metaheuristic algorithm, where their performance is evaluated using a predefined objective function.
3. **Elitism:** The top-performing individuals, based on fitness scores, are preserved in the next generation to ensure the retention of high-quality solutions.
4. **Variation Operators:** The remaining individuals undergo variation through crossover and mutation, with parent selection guided by a roulette-wheel selection mechanism. Structural constraints are applied to maintain the syntactic and semantic validity of the expressions.
5. **Fitness Evaluation:** The fitness of the new individuals is recalculated by integrating them into the algorithm and re-evaluating their performance.
6. **Termination:** Steps 2 to 5 are repeated until the stopping criterion is met, such as reaching the maximum number of generations or achieving a satisfactory performance level.

Algorithm 1 outlines the workflow of the GP-MAs framework. Inspired by prior studies on evolutionary formula generation in metaheuristics, this approach ensures structural diversity at initialization via a balanced tree construction strategy. Each individual, represented as a candidate metaheuristic rule, is transformed into an executable formula and evaluated within a host algorithm. Top individuals are retained via elitism, while the remaining ones undergo crossover and mutation under structural constraints. The evolution proceeds iteratively until convergence or reaching the iteration limit.

Algorithm 1: GP-MAs: genetic programming-based metaheuristic automation

Input: PS, $MAXD$, $MAXIT$, Evaluation function F, Function set, Terminal set
Output: Optimal evolved update rule
Initialization:
  - population POP^0 of size PS:
    - 50% via depth-first tree generation
    - 50% via breadth-first tree generation
    - Nodes drawn from function and terminal sets
Parse individuals (in-order) → executable expressions
Evaluate fitness of each individual using F
$iter \leftarrow 1$
while $iter \leq MAXIT$ **do**
    Preserve top 20% of POP^{iter} into POP^{iter+1} *(elitism)*
    Select parents via hybrid (roulette + tournament) selection
    crossover: subtree swapping
    mutation:
      - Node mutation
      - Subtree mutation
    evaluate offspring using F
    Add valid offspring to POP^{iter+1} until $|POP| = PS$
    $iter \leftarrow iter + 1$
end while
Return: Best-performing individual in final generation

To avoid excessive growth of GP trees (bloat), the maximum tree depth was restricted during the evolutionary process according to the predefined depth constraints in Table 1. Structural constraints were also applied during crossover and mutation to maintain syntactic validity and control expression complexity. Since the GP evolution stage is performed offline, the additional computational overhead is incurred only once during strategy generation and does not affect the subsequent online optimization stage. The overall computational complexity of GP-MAs can be approximated as $O(G_{GP} \times P_{GP} \times N_f \times R \times FEs \times C_f)$, where G_{GP} denotes the number of GP generations, P_{GP} is the population size, N_f represents the number of training functions, R is the number of independent runs, and C_f denotes the computational cost of fitness evaluation. In addition, protected division and bounded nonlinear operations were adopted during GP evolution to improve numerical stability and reduce the risk of singularities or excessively large update values caused by nested trigonometric operators.

To quantitatively assess the performance of evolved formulas, a customized evaluation function F is designed for the GP-MAs framework. To provide representative optimization landscapes during GP evolution, four benchmark functions are selected from the CEC2022 benchmark suite: F1 (unimodal), F3 (basic multimodal), F7 (hybrid), and F10 (composition). These functions were selected to represent different optimization characteristics, including unimodal, multimodal, hybrid, and composition landscapes, thereby improving the diversity of training scenarios during GP evolution. Each evolved formula is integrated into a metaheuristic algorithm variant, referred to as *new-MAs*, and tested on these functions.

Let fit_j be the mean result of *new-MAs* on function j over five independent runs, and fit_j^* denote the known global optimum. The overall fitness score is computed as:

$$\text{Fitness} = \sum_{j\in\{1,3,7,10\}} \frac{\text{fit}_j}{\text{fit}_j^*} \tag{21}$$

This aggregated metric provides an indirect assessment of the transferability and adaptability of evolved strategies across representative optimization scenarios.

Although the chosen benchmark functions include representative unimodal, multimodal, hybrid, and composition landscapes, the current study does not address extremely dynamic, limited, or non-stationary real-world optimization contexts. As a result, the current studies should be interpreted as representative validation rather than universal generalization verification. Future research will focus on more difficult optimization scenarios. The computational complexity of GP-MAs is dominated by GP evolution, which runs at O(G × P × F), where G is the number of generations, P is population size, and F is the fitness evaluation cost. Because GP evolution is performed offline, its cost has no impact on the runtime complexity of GPbasedGO during optimization.

3.3 The GP-GO Algorithm

To minimize numerical instability and overly complicated structures, evolving expressions are controlled during GP search with protected operators, bounded depth control, and fitness-based pruning. Only expressions that meet predefined complexity constraints are kept. This study proposes **GPbasedGO**, an upgraded form of the Growth Optimizer (GO) algorithm optimized using the GP-MAs framework. Within this paradigm, GP-MAs independently evolve the learning rule. The evaluation function F is set to a size of 10. In accordance with recognized techniques, four sample functions (F1, F3, F7, and F10) from the CEC2022 benchmark are chosen for evaluation.

The terminal set $\mathcal{T}$ and function set $\mathcal{F}$ critically determine GP performance. For GO optimization, these sets are designed as:

$$\begin{cases} \mathcal{T} = \{Gap_1, Gap_2, Gap_3, Gap_4, SF\} \\ \mathcal{F} = \{+, -, \times, \div, \text{norm}, \sin, \cos, \tan\} \end{cases}$$

To evaluate the contribution of GP-evolved operators, several ablation variants are defined: GO-original: baseline Growth Optimizer; GO-SF: only self-perception factor enabled; GO-fixed-rule: handcrafted update rule; GPbasedGO: full proposed model;

$\mathcal{T}$ encodes social hierarchy dynamics: Gap_{1-4} quantify rank disparities (e.g., leader-elite, elite-lower, random pairs) to guide knowledge acquisition through hierarchical learning mechanisms. SF (self-perception factor) dynamically scales learning intensity via growth resistance (GR), balancing local refinement ($SF \ll 1$) and global exploration ($SF \gg 1$). This adaptive scaling mirrors parameter self-adaptation strategies in differential evolution.

$\mathcal{F}$ integrates arithmetic operators for evolutionary recombination and *norm* for numerical stability. Trigonometric functions inject nonlinear perturbations to escape local optima while maintaining diversity. This hybrid design balances deterministic hierarchical learning with stochastic exploration, accelerating cnvergence while preserving solution diversity. Embedded self-adaptive components reduce parameter sensitivity while maintaining evolutionary pressure toward optimal regions, aligning with the "learning-reflection-self-improvement" philosophy of human-inspired optimization.

The update rule for this process can be expressed as follows:

$$\begin{aligned}\text{Term}_1 &= \left\|\text{Gap}_1\right\| \cdot \text{Gap}_1 + \left\|\text{Gap}_2\right\| \cdot \text{Gap}_2,\\ \text{Term}_2 &= \left\|\text{Gap}_3\right\| \cdot \text{Gap}_3\\ &\quad + \left\| \left(\text{Gap}_1 - \text{Gap}_1\right) - \left(SF \cdot SF\right) \right\| \cdot \text{Gap}_4\end{aligned}$$

The numerator of the update formula can then be written as:

$$\text{Numerator} = SF \cdot \left(\text{Term}_1 + \text{Term}_2\right). \tag{22}$$

The denominator is given by:

$$\begin{aligned}\text{Denominator}_1 &= \left\| \cos\left(\tan\left(\cos(\left\|\text{Gap}_1\right\|)\right)\right)\right\| + \left\|\text{Gap}_2\right\|,\\ \text{Denominator}_2 &= \left\|\text{Gap}_3\right\| + \left\| \sin\left(\left\| \cos(\text{Gap}_3 - \text{Gap}_2)\right\|\right)\right\|,\end{aligned} \tag{23}$$

Thus, the complete update equation becomes:

$$\text{Update} = \frac{\text{Numerator}}{\text{Denominator}_1 + \text{Denominator}_2}. \tag{24}$$

Compared with the original formulation, the new update equation is automatically evolved through Genetic Programming (GP), enabling more adaptive information interaction among different gap terms. Specifically, Gap_1 and Gap_2 mainly emphasize elite-guided exploitation, while Gap_4 introduces stochastic diversity to enhance global exploration. The self-perception factor SF adaptively adjusts the learning intensity according to the evolutionary state of individuals. In addition, the nonlinear operators, including trigonometric functions and normalization terms, increase directional diversity and help maintain the balance between exploration and exploitation during the search process. The denominator terms further normalize the update magnitude, which contributes to numerical stability and avoids excessively large search steps. Therefore, compared with the original manually designed formulation, the evolved symbolic update rule provides a partially interpretable adaptive learning mechanism for complex optimization tasks.

3.4 Fusion Method Based on GPbasedGO Algorithm

To further improve the performance of image fusion, the basic fusion formula (Eq. (10)) is modified, and the updated formulation is expressed in Eq. (25):

$$F_i = M_i + g_i \Gamma_i (P - I) \tag{25}$$

where Γ_i denotes a weight matrix that plays a pivotal role in determining the relative contributions of the panchromatic image P and the low-resolution multispectral image M_i during the fusion process, thereby directly influencing the quality of the fused output. The definition of Γ_i is given in Eq. (26):

$$\Gamma_i = \beta(i)\Gamma_P + \big(1 - \beta(i)\big)\Gamma_M \tag{26}$$

where $\beta(i)$ is a coefficient associated with the i-th spectral band of the multispectral image. By adjusting $\beta(i)$, the relative proportions of Γ_P and Γ_M within Γ_i can be flexibly controlled. The computation of Γ_P and Γ_M is described in Eqs. (27) and (28), respectively:

$$\Gamma_P = \exp\left(-\frac{\lambda}{\|\nabla P\|^4 + \varepsilon}\right) \tag{27}$$

$$\Gamma_M = \exp\left(-\frac{\lambda}{\|\nabla M_i\|^4 + \varepsilon}\right) \tag{28}$$

where ∇P and ∇M_i denote the gradients of the panchromatic and multispectral images, respectively. The gradient encodes pixel intensity variations, enabling the preservation of edge and detail features during fusion. Incorporating gradient information thus facilitates more precise processing in the fusion stage. The term ε is a small constant introduced to prevent division by zero, ensuring numerical stability and accuracy. The denominators $\|\nabla P\|^4 + \varepsilon$ and $\|\nabla M_i\|^4 + \varepsilon$ are safeguarded against zero values. The parameter λ governs the degree of smoothness in the resulting fused image. Empirical evaluation demonstrates that setting $\lambda = 10^{-9}$ and $\varepsilon = 10^{-10}$ yields substantial improvements in spectral resolution. Under this configuration, the fused image more faithfully preserves spectral characteristics across different bands, thus enhancing spectral fidelity and meeting the high-quality imaging requirements of remote sensing and related applications.

Through the proposed definition of Γ_i and careful parameter tuning, the integration of multispectral and panchromatic imagery can be performed more effectively, producing high-quality fusion results. This provides superior input data for subsequent remote sensing analysis and processing tasks. The overall workflow of the proposed GP-GO-based remote sensing image fusion approach is illustrated in Fig. 6.

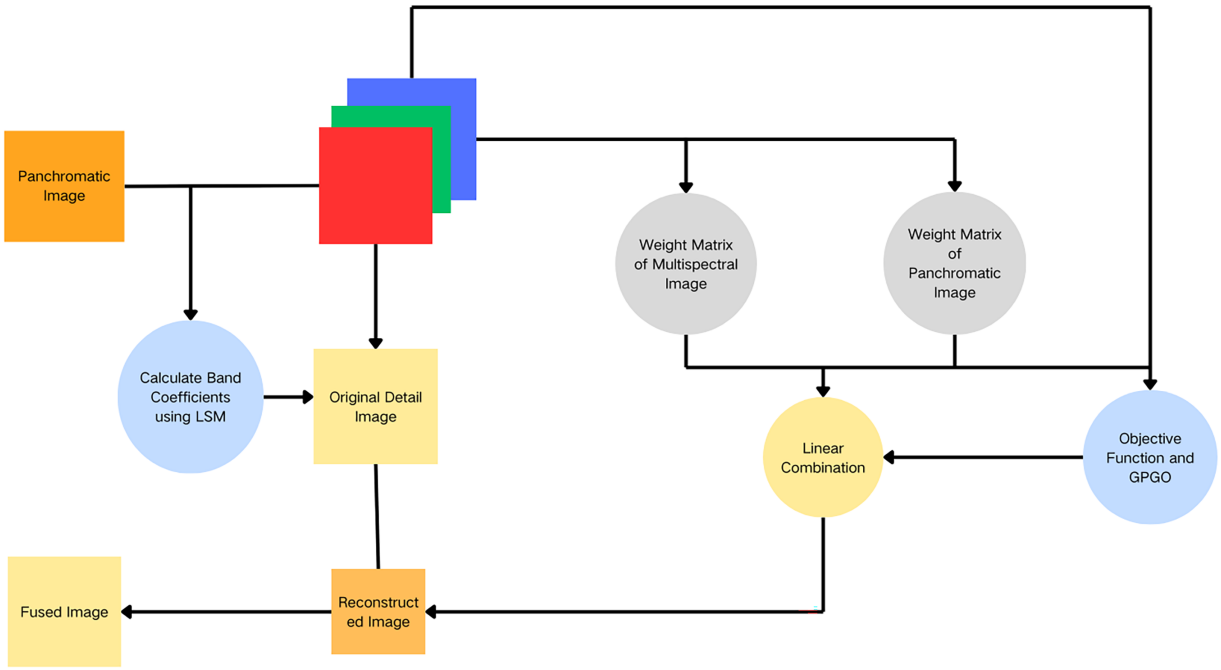


Figure 6: Flowchart of the proposed fusion method

4 Experimental Results

This section describes the experimental framework for testing the proposed GP-based GO algorithm. Representative benchmark functions from the CEC2022 test suite are used to evaluate optimization performance in difficult settings. A remote sensing image fusion task (panchromatic-multispectral fusion) further validates the practical efficacy of GP-basedGO. This dual-assessment framework analyzes robustness, adaptability, and optimization skills in both synthetic and real-world environments.

4.1 Experimental Setup

A comprehensive comparative study is conducted against fifteen algorithms, comprising classical methods (PSO, GA, ASO, BOA, SA, CS, GWO, DE, CRO) and recent state-of-the-art metaheuristics (GO, WOO [50], EMWPA [51], FNO [52], CHO [53], LEE [54]). To maintain fairness, all algorithms use the same population sizes and fixed fitness evaluations (FES). Each algorithm is run 30 times individually on each benchmark function to mitigate the effect of initialization randomization on solution accuracy and assure statistical reliability. The performance is evaluated using the mean and standard deviation of the answers, as well as the Wilcoxon rank-sum test. All experiments in this study were carried out on a Windows 11 computer platform outfitted with an NVIDIA GeForce GTX 1650 GPU with 4 GB of video RAM and CUDA parallel computing capabilities. The central processing unit (CPU) is an Intel Core i7-9750H processor with six cores running at 2.60 GHz, and the system comes with 16 GB of RAM.

To assess the performance of the proposed GPbasedGO algorithm, the standard CEC2022 benchmark suite was used. This test set includes four types of functions: unimodal, basic multimodal, hybrid, and composition functions. The unimodal function (F1) is specifically developed to evaluate the algorithm's local search capabilities and convergence accuracy because it only has one global optimum.

The basic multimodal functions (F2–F5) contain a significant number of local optima, which are used to assess the algorithm's global exploration capacity and ability to avoid premature convergence to local minima. These roles present a considerable challenge in terms of striking an efficient balance between exploration and exploitation.

The hybrid functions (F6–F8) are created by merging various basic functions with different properties, resulting in more complex and deceiving landscapes. These functions test the algorithm's robustness and flexibility in dynamic and heterogeneous optimization contexts.

Finally, the composition functions (F9–F12) are the most complex category. They are created by combining many hybrid or basic functions with different biases, rotations, and scaling transformations. These functions add considerable levels of nonlinearity, modality, and search space irregularity, serving as a demanding testbed for evaluating the algorithm's capacity to locate and transition between numerous diverse regions of the search space.

Through extensive experimentation across this broad range of benchmark functions, the GPbasedGO algorithm's effectiveness and generalization potential may be comprehensively tested in optimization situations of different difficulty and landscape complexity.

4.1.1 Analysis of 10-Dimensional Results

Tables 2–4 compare the GP-basedGO method to fifteen different optimization strategies for 10-dimensional problems. The comparison of mean values demonstrates that GP-basedGO has greater overall performance. The statistics demonstrate that GP-basedGO generally outperforms other algorithms, with its mean metric exhibiting statistically significant improvements on a wide range of test functions, particularly F1, F4, F6, and F11. However, for some functions, like as F3 and F7, a few rival algorithms may perform similarly to or slightly better than GPbasedGO.

Table 2: 10-dimensional comparison results (Part 1).

Func_Num	Mean_PSO	R	Mean_GA	R	Mean_DE	R	Mean_GWO	R	Mean_GO	R	Mean_GPbasedGO
1	8828.213	+	7178.107	+	4653.567	+	5883.246	+	2070.409	+	761.238
2	481.4768	+	891.4384	+	421.6076	+	443.5214	+	408.641	+	403.9103
3	622.0307	+	641.7936	+	608.4225	+	602.7519	+	601.6679	=	601.4074
4	856.9073	+	841.992	+	850.7588	+	822.5716	=	838.2599	+	821.5802
5	1274.636	+	1237.221	+	938.7871	+	946.3514	=	926.8552	+	914.3811
6	4344257	+	56128305	+	17187.74	+	21769.89	+	28638.76	+	4487.426
7	2070.12	+	2078.786	+	2052.423	+	2049.472	+	2035.152	+	2023.685
8	2237.394	+	2248.042	+	2231.638	+	2263.761	+	2228.674	+	2224.31
9	2587.122	+	2704.782	+	2506.232	=	2593.41	+	2529.35	+	2501.057
10	2624.657	+	2547.583	=	2546.931	=	2591.171	=	2547.91	=	2557.738
11	2995.133	+	3080.935	+	2828.305	+	2828.417	=	2689.166	=	2749.009
12	2875.162	+	3001.373	+	2862.248	+	2869.924	+	2863.979	+	2860.399
Win		12		11		11		8		9	

Table 3: 10-dimensional comparison results (Part 2).

Func_Num	Mean_ASO	R	Mean_BOA	R	Mean_SA	R	Mean_CS	R	Mean_CRO	R	Mean_GPbasedGO
1	11718.16	+	14424.11	+	10635.12	+	20780.83	+	2331248	+	761.238
2	489.1466	+	2221.291	+	526.9067	+	1651.528	+	4592.578	+	403.9103
3	620.5264	+	668.0617	+	649.047	+	668.4357	+	699.9431	+	601.4074
4	838.7514	+	875.0021	+	874.2788	+	890.413	+	928.6312	+	821.5802
5	1041.59	+	1787.522	+	1528.244	+	2680.351	+	4934.607	+	914.3811
6	1004955	+	2.58E+08	+	4323.045	=	1.56E+08	+	1.62E+09	+	4487.426
7	2087.336	+	2123.092	+	2106.302	+	2144.093	+	2274.834	+	2023.685
8	2240.496	+	2317.531	+	2249.618	+	2313.266	+	3270.592	+	2224.31
9	2652.427	+	2823.84	+	2724.674	+	2819.166	+	3176.525	+	2501.057
10	2517.251	=	2859.451	+	2756.895	+	2738.914	+	4144.509	+	2557.738
11	2796.109	+	3811.853	+	3111.818	+	3743.182	+	5064.099	+	2749.009
12	2927.294	+	3085.632	+	2963.107	+	2977.527	+	3238.392	+	2860.399
Win		11		12		11		12		12	

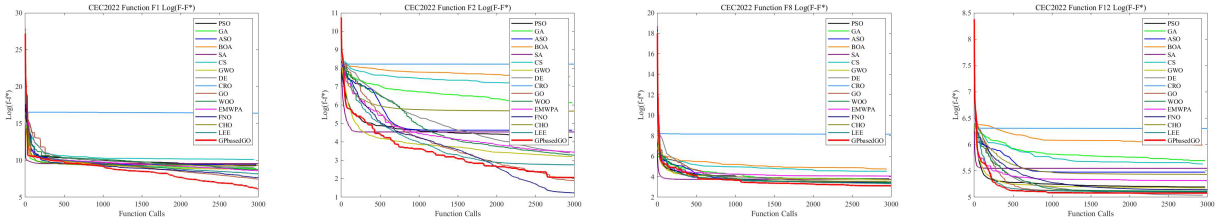
Table 4: 10-dimensional comparison results (Part 3).

Func_Num	Mean_WOO	R	Mean_EMWPA	R	Mean_FNO	R	Mean_CHO	R	Mean_LEE	R	Mean_GPbasedGO
1	6536.274	+	5690.298	+	2317.047	+	11615.56	+	3841.837	+	761.238
2	426.5368	+	431.3476	+	403.4368	=	689.9362	+	415.7015	+	403.9103
3	614.2211	+	631.5148	+	601.7645	=	623.9042	+	601.866	+	601.4074
4	842.7871	+	832.3918	+	819.6097	=	828.5045	=	821.1671	=	821.5802
5	1040.472	+	1148.165	+	905.6369	=	1099.184	+	903.5064	=	914.3811
6	291645.1	+	3238.29	=	3255.117	=	28586566	=	20309.67	+	4487.426
7	2052.795	+	2074.466	+	2025.991	=	2064.781	+	2036.688	+	2023.685
8	2231.118	+	2258.296	+	2228.317	+	2234.951	+	2228.119	+	2224.31
9	2547.921	+	2569.291	+	2531.678	+	2672.473	+	2529.386	+	2501.057
10	2505.501	–	2666.879	+	2535.403	=	2599.365	+	2519.944	=	2557.738
11	2785.318	+	2899.059	+	2663.504	–	3111.296	+	2669.862	–	2749.009
12	2866.585	+	2903.486	+	2871.114	+	2928.741	+	2863.776	+	2860.399
Win		11		11		7		12		8	

When GPbasedGO is compared to classical algorithms (such as PSO, GA, and DE), its distinct advantages become clear. According to the tables, GP-basedGO outperforms other methods in a variety of test conditions. When compared to PSO, GP-basedGO has smaller mean values and faster convergence across functions, indicating improved performance in optimizing high-dimensional complex landscapes. When compared to GA, GP-basedGO has a substantially lower standard deviation, indicating improved stability, precision, and reduced sensitivity to initial circumstances and random causes. GP-basedGO outperforms DE and related algorithms in terms of multimodal function search. Furthermore, its advantages over algorithms such as GWO and CS are essentially based on balancing unimodal and multimodal optimization, demonstrating its strong problem-solving abilities for complicated problems.

Furthermore, as compared to current state-of-the-art metaheuristics, GP-basedGO maintains a significant competitive advantage. In comparison to high-performing approaches such as CHO and EMWPA, GP-basedGO outperforms them on complicated multimodal landscapes, where it more successfully avoids local optima to ensure higher precision. While recently presented algorithms such as FNO and LEE excel at certain, isolated functions, GP-basedGO provides a far more balanced and versatile performance profile over the whole benchmark suite, demonstrating its broad application.

Fig. 7 shows the convergence curves of various methods in a 10-dimensional space. The picture depicts representative portions of convergence behavior, with subplot names identifying the relevant benchmark functions. It is clear that GP-basedGO (shown by the red curve) has strong optimization capabilities, with a constant downward trajectory that eventually outperforms practically all other algorithms. Most algorithms converge quickly within the first 500 evaluations; however, on complex functions like F2 and F12, competitors such as CS and CRO tend to plateau early on. However, GP-basedGO well balances local exploitation and global exploration, which is an important property for optimizing complex systems. This equilibrium avoids premature convergence to local optima while efficiently traversing the whole search space to find high-quality solutions.



(**a**) F1 (**b**) F2 (**c**) F8 (**d**) F12

Figure 7: 10-dimensional convergence plots of the algorithm.

4.1.2 Analysis of 20-Dimensional Results

Tables 5–7 compare the 20-dimensional outcomes of the GP-basedGO method to fifteen other optimization algorithms. Notably, the GP-based GO updating approach was developed with only 10-dimensional training data. Its higher performance on 20-dimensional test functions reveals high dimensionality transferability, implying that the evolved formula reflects scalable optimization processes rather than simply overfitting the low-dimensional training set. The average values in the tables show that GPbasedGO outperforms other methods on the majority of functions. On specific test functions such as F1, F2, F8, and F12, GP-basedGO outperforms other algorithms in terms of average value metrics. However, for some functions, such as F10, a few rival algorithms may perform similarly or slightly better than GPbasedGO.

Table 5: 20-dimensional comparison results (Part 1).

Func_Num	Mean_PSO	R	Mean_GA	R	Mean_DE	R	Mean_GWO	R	Mean_GO	R	Mean_GPbasedGO
1	52904.89	+	38427.22	+	54598.73	+	30783.65	+	19378.57	=	18987.99
2	791.1697	+	1435.08	+	616.9638	+	531.2367	+	473.6368	=	471.8706
3	636.9382	+	675.4038	+	630.5062	+	614.387	=	609.4576	=	611.4177
4	968.9801	+	949.5482	+	957.3419	+	864.6567	=	932.2348	+	868.1742
5	3389.725	+	3130.209	+	2040.769	+	1514.343	=	1447.686	=	1371.278
6	3.16E+08	+	7.93E+08	+	25570941	+	3841266	+	2683454	+	146310.6
7	2196.214	+	2180.967	+	2225.742	+	2118.254	=	2134.449	+	2095.174
8	2334.213	+	2429.743	+	2317.251	+	2275.405	+	2251.454	+	2244.28
9	2620.435	+	3059.033	+	2519.274	+	2561.716	+	2484.558	+	2474.834
10	5862.466	+	3977.858	=	5745.673	+	4394.218	+	3732.7	=	3577.976
11	4637.618	+	7291.082	+	4381.296	+	4010.602	+	3239.342	=	3267.323
12	3097.133	+	3753.875	+	2900.005	–	3027.249	+	2961.031	+	2915.146
Win		12		11		11		8		6	

Table 6: 20-dimensional comparison results (Part 2).

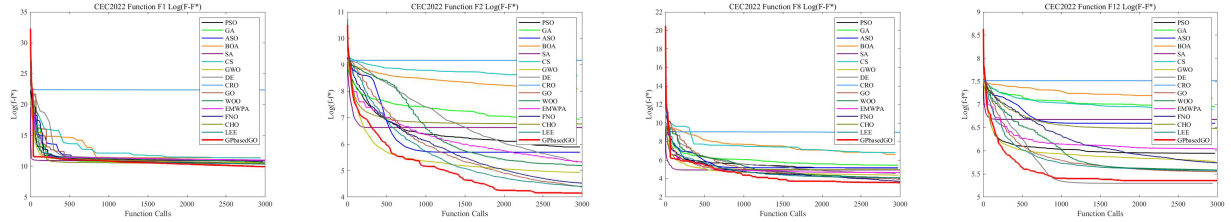
Func_Num	Mean_ASO	R	Mean_BOA	R	Mean_SA	R	Mean_CS	R	Mean_CRO	R	Mean_GPbasedGO
1	47506.21	+	99751.94	+	56751.55	+	82307.3	+	1.28E+09	+	18987.99
2	704.7262	+	3692.529	+	1069.681	+	5435.199	+	8923.236	+	471.8706
3	644.7519	+	697.827	+	692.8078	+	708.7662	+	738.0695	+	611.4177
4	927.8916	+	1020.596	+	1025.24	+	1064.253	+	1108.797	+	868.1742
5	2416.787	+	5299.146	+	3482.977	+	9737.955	+	14244.68	+	1371.278
6	11609227	+	2.75E+09	+	7888.752	–	3.44E+09	+	8.79E+09	+	146310.6
7	2242.773	+	2301.811	+	2227.635	+	2320.135	+	2535.034	+	2095.174
8	2359.021	+	2909.087	+	2324.389	+	2903.279	+	31644.07	+	2244.28
9	2688.301	+	3675.949	+	2892.854	+	3409.789	+	4881.742	+	2474.834
10	4580.649	=	6886.978	+	4959.543	+	6572.561	+	8394.181	+	3577.976
11	4633.79	+	9374.592	+	7549.17	+	11243.98	+	15367.25	+	3267.323
12	3479.162	+	4029.731	+	3527.145	+	3676.27	+	4493.905	+	2915.146
Win		11		12		11		12		12	

Table 7: 20-dimensional comparison results (Part 3).

Func_Num	Mean_WOO	R	Mean_EMWPA	R	Mean_FNO	R	Mean_CHO	R	Mean_LEE	R	Mean_GPbasedGO
1	43791.76	+	60062.51	+	31885.35	+	35319.07	+	31636.37	+	18987.99
2	580.321	+	606.6398	+	492.6074	+	1268.704	+	481.9139	=	471.8706
3	630.0162	+	658.2631	+	613.9394	+	651.1437	+	607.1674	–	611.4177
4	927.687	+	901.6484	+	887.4723	+	931.0341	+	881.7019	+	868.1742
5	1977.039	+	2829.912	+	1347.751	=	2773.033	+	1185.675	–	1371.278
6	22469984	+	520960	=	19478.4	–	3.73E+08	+	742251.9	+	146310.6
7	2159.721	+	2183.707	+	2118.613	+	2196.076	+	2135.334	+	2095.174
8	2260.329	+	2305.105	+	2239.821	+	2342.964	+	2259.323	+	2244.28
9	2525.473	+	2563.305	+	2497.094	+	2906.536	+	2487.576	+	2474.834
10	3546.754	=	4341.949	+	3132.283	=	3999.009	=	3775.509	=	3577.976
11	4418.292	+	3766.687	+	3113.655	–	6980.613	+	3113.645	–	3267.323
12	2966.472	+	3119.486	+	3011.42	+	3355.557	+	2963.818	+	2915.146
Win		12		12		9		12		9	

When compared to standard algorithms (e.g., PSO, GA, DE, GWO, and GO), GP-basedGO has distinct advantages and beats them in stability. When compared to PSO, GPbasedGO has lower average values on most functions and faster convergence in the initial stage, indicating improved performance in high-dimensional complex function optimization. In comparison to GA, GP-basedGO has a lower standard deviation, indicating more stable results with less effect from initial conditions and random causes. GP-basedGO outperforms DE in terms of multimodal function search. Furthermore, it beats algorithms like as GWO and GO in both unimodal and multimodal settings. Furthermore, GP-basedGO remains very competitive with newer cutting-edge metaheuristics (e.g., WOO, FNO), indicating its robust capacity to tackle complicated problems in a variety of terrains.

Fig. 8 shows the convergence curves of various methods in a 20-dimensional space. We offer only representative parts of the convergence graphs here, with subplot names specifying the corresponding benchmark functions. As illustrated by the convergence curves, the move to 20 dimensions causes many algorithms to plateau due to the "curse of dimensionality" (e.g., CS and CRO on specific functions). However, GPbasedGO has a consistent decreasing tendency, displaying significant optimization abilities and surpassing almost all other algorithms in the end results. This robustness comes from the GP-evolved formula's inherent scalability, which uses non-linear operators to generate various step sizes, eliminating diversity loss and assuring robust performance in higher-dimensional spaces. GPbasedGO iterates between two phases, each having a growth cycle, allowing it to efficiently balance local and global exploration. During the seedling growth stage, the population update approach slows and stabilizes the search space exploration process. This strategy allows for some exploration of new regions while focusing more on the in-depth evolution of the surrounding environment, preventing premature convergence to local optima and enabling extremely effective search space utilization.



(**a**) F1 (**b**) F2 (**c**) F8 (**d**) F12

Figure 8: 20-dimensional convergence plots of the algorithm.

4.2 Experimental Evaluation of GPbasedGO on Multispectral and Panchromatic Image Fusion

This section looks at the performance and applicability of the Genetic Programming-based Growth Optimizer method (GPbasedGO) for adaptive multispectral and panchromatic image fusion applications. To test the usefulness of GP-basedGO in tackling such fusion difficulties, benchmark experimental data from two satellites with different theme characteristics are used. The publicly available NBUPansharpRSData dataset is used, which can be found at https://github.com/starboot/NBUPansharpRSData. This comprehensive dataset contains multispectral photos acquired by satellites including QuickBird, IKONOS, GF, and WorldView.

Four sample scenarios from the popular QuickBird and GF satellite pictures are chosen for focused and representative examination. The experimental procedure is organized in the following five topic modules:

1. **Dataset and Parameter Configuration:** Description of the remote sensing data characteristics and algorithm parameter settings.

2. **Urban Scene Analysis:** Performance comparison of 15 state-of-the-art fusion algorithms in urban landscape scenarios.
3. **Vegetation Scene Analysis:** Quantitative evaluation of 15 benchmark algorithms in green vegetation areas.
4. **Aquatic Environment Analysis:** Objective assessment of 15 established methods in water-dominated environments.
5. **Coastal Region Analysis:** Comprehensive comparison of 15 representative algorithms in coastal terrain conditions.

4.2.1 Experiments in Urban Scenarios

Table 8 shows that the suggested GP-based GO consistently beats all four urban settings. In *QuickBird (Image 1)*, GPbasedGO gets the best values in RASE (20.39585), RMSE (12.63616), UIQI (0.976627), and CC (0.970901), while maintaining an ERGAS value of 382.4032, which is extremely close to the optimal result attained by EMWPA. Although EMWPA outperforms GP-basedGO in ERGAS and SAM, the proposed technique has greater spatial-spectral consistency, as evidenced by higher UIQI and CC values.

Table 8: Experimental results of sixteen algorithms for urban scenarios.

Satellite	Algorithm	ERGAS	SAM	RASE	RMSE	UIQI	CC
QuickBird (Image1)	GPbasedGO	382.4032	5.243785	20.39585	12.63616	0.976627	0.970901
	GO	382.6416	5.285517	20.91623	12.95856	0.975449	0.968729
	PSO	382.4959	5.252925	20.58237	12.75172	0.976208	0.970186
	GA	382.644	5.314749	20.55184	12.7328	0.976276	0.970248
	ASO	382.7086	5.339281	20.60324	12.76465	0.97616	0.970041
	BOA	382.5004	5.23277	20.73429	12.84584	0.975865	0.969473
	SA	382.8712	5.342528	20.67597	12.80971	0.975997	0.969727
	CS	382.542	5.370759	20.86571	12.92726	0.975564	0.968902
	GWO	382.5662	5.324692	20.45936	12.67551	0.976485	0.970622
	DE	382.8932	5.371044	20.85456	12.92035	0.97559	0.969017
	CRO	382.4874	5.271182	20.54773	12.73026	0.976285	0.970252
	CHO	382.5125	5.223818	20.5866	12.75434	0.976199	0.970113
	EMWPA	382.2927	5.189105	20.70135	12.82543	0.97594	0.969652
	FNO	382.4804	5.321729	20.50633	12.70461	0.97638	0.970446
	LEE	382.5837	5.346564	20.52312	12.71501	0.976341	0.970398
	WOO	382.5801	5.433496	20.75027	12.85574	0.975827	0.969551
QuickBird (Image2)	GPbasedGO	25.39373	19.06324	101.2538	64.52623	0.65957	0.828362
	GO	27.77494	19.07814	109.964	70.07693	0.621415	0.752443
	PSO	42.33413	19.15463	168.1802	107.1765	0.41323	0.701326
	GA	30.37785	19.21056	120.9853	77.10051	0.576064	0.797326
	ASO	26.3722	19.08764	105.1434	67.00493	0.642176	0.781801
	BOA	27.44076	19.14416	110.0986	70.16276	0.620876	0.751387
	SA	31.63866	19.10426	125.8375	80.1927	0.556758	0.781697
	CS	31.50741	19.17038	125.0475	79.68922	0.559709	0.755789
	GWO	25.71951	19.11131	102.5428	65.34764	0.653889	0.825395
	DE	39.67934	19.13867	157.6182	100.4457	0.444955	0.750105
	CRO	27.63907	19.07752	109.452	69.75071	0.62353	0.748733
	CHO	26.63924	19.06961	105.6608	67.33463	0.639994	0.78389

(Continued)

Table 8 (continued)

Satellite	Algorithm	ERGAS	SAM	RASE	RMSE	UIQI	CC
	EMWPA	27.02578	19.1105	107.5041	68.50933	0.631942	0.764356
	FNO	27.33127	19.10924	109.0162	69.47296	0.625773	0.808846
	LEE	28.05354	19.19389	111.3328	70.94928	0.615635	0.76044
	WOO	26.88133	19.11495	107.6847	68.62444	0.631164	0.765998
GF(Image1)	GPbasedGO	13.45477	7.445838	27.88895	29.02449	0.922608	0.985052
	GO	13.46419	7.526927	27.96688	29.10559	0.922164	0.984017
	PSO	13.45623	7.470751	27.90803	29.04435	0.922519	0.984751
	GA	13.47017	7.515901	27.96922	29.10803	0.922143	0.984183
	ASO	13.46013	7.513942	27.94635	29.08423	0.922288	0.984234
	BOA	13.46837	7.487018	27.93651	29.07398	0.922345	0.984681
	SA	13.45749	7.482417	27.92112	29.05797	0.922419	0.984635
	CS	13.45694	7.465802	27.90467	29.04085	0.922507	0.984892
	GWO	13.46231	7.493255	27.96188	29.10039	0.922184	0.984101
	DE	13.46666	7.505501	27.9468	29.0847	0.922281	0.98446
	CRO	13.46996	7.464909	27.92497	29.06198	0.922393	0.984954
	CHO	13.46455	7.481235	27.93293	29.07026	0.922337	0.984675
	EMWPA	13.47318	7.471677	27.93535	29.07278	0.922362	0.984866
	FNO	13.46659	7.483574	27.93991	29.07753	0.9223	0.984604
	LEE	13.45749	7.471695	27.91282	29.04934	0.92247	0.984742
	WOO	13.46043	7.489816	27.93241	29.06973	0.922342	0.984546
GF (Image2)	GPbasedGO	14.03216	8.827921	33.92236	30.91043	0.898796	0.974964
	GO	16.12653	8.908509	48.41519	44.11646	0.812654	0.831852
	PSO	17.53988	8.942428	46.39015	42.27122	0.825565	0.879807
	GA	18.64609	8.926267	52.89738	48.20068	0.784312	0.837841
	ASO	16.07145	8.989749	48.27734	43.99085	0.813275	0.796982
	BOA	15.09571	8.927943	36.64079	33.3875	0.883811	0.967579
	SA	15.84673	8.919543	41.85795	38.14143	0.853205	0.898423
	CS	14.82983	8.897199	38.65229	35.2204	0.872155	0.918336
	GWO	15.13084	9.009675	36.93867	33.65893	0.882022	0.961266
	DE	15.96863	9.015953	39.84708	36.3091	0.865238	0.941227
	CRO	17.24998	8.904977	52.69489	48.01617	0.785549	0.822924
	CHO	14.99604	8.889481	36.24556	33.02736	0.886059	0.97098
	EMWPA	16.02911	8.950923	43.49184	39.63025	0.843217	0.872788
	FNO	15.22236	8.988833	43.3377	39.4898	0.844036	0.84813
	LEE	17.63411	8.910252	53.8285	49.04913	0.77848	0.827633
	WOO	15.08025	8.897155	40.95523	37.31886	0.858568	0.890164

In the challenging *QuickBird (Image 2)* scene, GP-basedGO beats all other approaches by achieving the best results on all six evaluation metrics: ERGAS, SAM, RASE, RMSE, UIQI, and CC. In particular, when compared to the original GO, the proposed method reduces ERGAS from 27.77494 to 25.39373 while increasing CC from 0.752443 to 0.828362, indicating that the introduced Gaussian-process-guided search strategy significantly improves the ability to preserve structural details in complex urban textures.

In both the *GF (Image 1)* and *GF (Image 2)* datasets, GPbasedGO consistently outperforms other approaches. GF (Image 2) outperforms all baselines, with significant decreases in ERGAS, RASE, and

RMSE, as well as the highest UIQI and CC values. These results show that the proposed technique may effectively balance spectral fidelity and spatial detail preservation despite differences in sensor qualities and scene complexity.

Visual comparisons in Fig. 9 demonstrate the efficacy of GP-based GO. The proposed method yields fused images with improved edge structures, richer building textures, and more natural color distributions, especially at road crossings, rooftops, and heavily populated regions. Compared to the low-resolution multispectral and panchromatic inputs, the fusion outputs have higher contrast and better spatial characteristics without causing significant spectral distortion. This observation supports the better quantitative scores in Table 8, indicating that GP-based GO provides robust and reliable fusion performance for urban remote sensing scenarios.

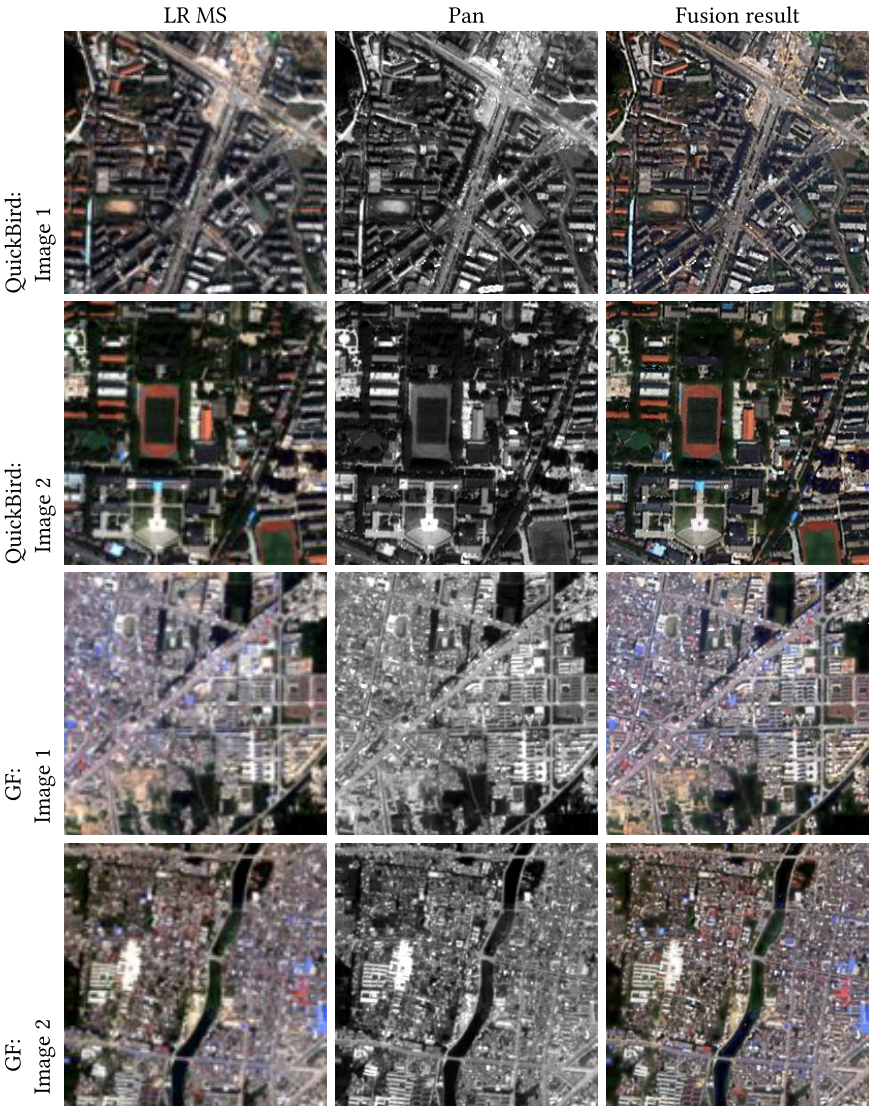


Figure 9: Visual assessment for marked region in Urban dataset.

4.2.2 Experiments in Green Vegetation Scenarios

Table 9 shows that the proposed GP-basedGO outperforms competing approaches in green vegetation conditions across four test cases. For *QuickBird (Image 1)*, GPbasedGO achieves the best values in five out of six evaluation measures, including ERGAS (15.68661), RASE (53.17873), RMSE (36.47474), UIQI (0.849425), and CC (0.985096), with a SAM value that is just somewhat inferior than that of EMWPA. This shows that the suggested technique is extremely effective in preserving vegetative structures while maintaining global spectral uniformity.

Table 9: Experimental results of sixteen algorithms for urban scenarios.

Satellite	Algorithm	ERGAS	SAM	RASE	RMSE	UIQI	CC
QuickBird (Image1)	GPbasedGO	15.68661	18.27238	53.17873	36.47474	0.849425	0.985096
	GO	15.69409	18.30768	53.20853	36.49518	0.849271	0.984935
	PSO	15.69512	18.34093	53.21522	36.49977	0.849242	0.984788
	GA	15.69176	18.33045	53.20147	36.49034	0.849323	0.984894
	ASO	15.69029	18.31001	53.19292	36.48448	0.849376	0.984999
	BOA	15.68712	18.27799	53.18109	36.47636	0.849409	0.98508
	SA	15.69426	18.32118	53.20925	36.49568	0.849268	0.984944
	CS	15.69383	18.33392	53.2063	36.49365	0.849398	0.984931
	GWO	15.69157	18.34527	53.20223	36.49086	0.849347	0.984863
	DE	15.71675	18.37958	53.32891	36.57775	0.848636	0.983516
	CRO	15.69536	18.32922	53.22345	36.50542	0.849186	0.984647
	CHO	15.70043	18.34868	53.23246	36.5116	0.849157	0.984829
	EMWPA	15.72472	18.27201	53.32528	36.57526	0.84861	0.984516
	FNO	15.69014	18.29638	53.19181	36.48371	0.849398	0.985019
	LEE	15.69263	18.27273	53.20553	36.49312	0.849282	0.984814
	WOO	15.69429	18.38283	53.2195	36.50271	0.849326	0.984578
QuickBird (Image2)	GPbasedGO	16.56209	14.44333	57.57588	29.08022	0.866672	0.954173
	GO	16.60697	14.60079	57.77096	29.17875	0.865889	0.95308
	PSO	16.58682	14.51683	57.67473	29.13015	0.866295	0.952936
	GA	16.59423	14.63537	57.70275	29.1443	0.866197	0.952377
	ASO	16.56317	14.50073	57.58074	29.08267	0.866644	0.953919
	BOA	16.58124	14.5296	57.65397	29.11966	0.86637	0.953785
	SA	16.61132	14.53013	57.78737	29.18704	0.865837	0.953088
	CS	16.58429	14.4658	57.71815	29.15207	0.866047	0.952784
	GWO	16.58584	14.53159	57.66733	29.12641	0.866382	0.953378
	DE	16.61269	14.51159	57.8871	29.23741	0.86532	0.951207
	CRO	16.59884	14.60928	57.70902	29.14746	0.866176	0.952118
	CHO	16.57462	14.60207	57.65369	29.11952	0.866313	0.953132
	EMWPA	16.59229	14.52568	57.76001	29.17322	0.865859	0.952379
	FNO	16.56999	14.58248	57.61374	29.09934	0.866516	0.953835
	LEE	16.58567	14.51641	57.71636	29.15117	0.866061	0.952694
	WOO	16.60701	14.68521	57.7587	29.17255	0.865961	0.951572

(Continued)

Table 9 (continued)

Satellite	Algorithm	ERGAS	SAM	RASE	RMSE	UIQI	CC
GF (Image1)	GPbasedGO	11.03369	18.01444	68.39465	32.51927	0.875023	0.98008
	GO	11.08113	18.24655	68.52269	32.58015	0.874619	0.979795
	PSO	11.18496	18.09741	68.81982	32.72142	0.873751	0.974931
	GA	11.05841	18.02943	68.42458	32.5335	0.874921	0.979624
	ASO	11.08614	18.21289	68.65736	32.64418	0.87433	0.979786
	BOA	11.08962	18.01497	68.51306	32.57557	0.874727	0.978261
	SA	11.12307	18.1807	68.71723	32.67264	0.874102	0.979474
	CS	11.10101	18.19416	68.54903	32.59267	0.874475	0.978983
	GWO	11.07836	18.20341	68.53413	32.58558	0.874585	0.979693
	DE	11.24237	18.18872	69.42385	33.00862	0.872011	0.976998
	CRO	11.0984	18.28245	68.63133	32.6318	0.874367	0.979385
	CHO	11.12918	18.13452	68.54482	32.59067	0.874458	0.978289
	EMWPA	11.082	18.17025	68.45601	32.54844	0.874742	0.979691
	FNO	11.07325	18.07449	68.55999	32.59788	0.874569	0.97924
	LEE	11.06608	18.22114	68.44944	32.54532	0.87482	0.979863
	WOO	11.10394	18.0813	68.52132	32.57949	0.874557	0.978485
GF (Image2)	GPbasedGO	4.97729	9.332181	30.7562	27.06278	0.955104	0.984087
	GO	5.003414	9.3741	30.78851	27.0912	0.955002	0.983678
	PSO	4.988541	9.403673	30.83979	27.13633	0.954855	0.984063
	GA	5.009212	9.355943	30.81154	27.11148	0.954941	0.983367
	ASO	5.022297	9.361167	30.79106	27.09345	0.954999	0.98354
	BOA	5.010704	9.480085	31.07598	27.34416	0.954179	0.984041
	SA	5.022346	9.350732	30.78092	27.08453	0.955039	0.983644
	CS	5.017558	9.37724	30.79101	27.09341	0.954998	0.98357
	GWO	5.013981	9.380036	30.78343	27.08674	0.955022	0.983857
	DE	5.021458	9.394264	30.80944	27.10962	0.954948	0.98365
	CRO	4.991599	9.430574	30.89768	27.18727	0.954684	0.983874
	CHO	5.001699	9.411901	30.85828	27.1526	0.954805	0.983883
	EMWPA	4.992035	9.343569	30.77741	27.08144	0.955039	0.98385
	FNO	4.979407	9.359269	30.76758	27.07279	0.955064	0.984034
	LEE	5.013111	9.371678	30.78751	27.09032	0.955006	0.983641
	WOO	5.010921	9.335741	30.78045	27.08412	0.955042	0.983708

In the tough *QuickBird (Image 2)* scene, GPbasedGO performs optimally on all six quality measures. Compared to the original GO, the proposed technique lowers ERGAS from 16.60697 to 16.56209 and SAM from 14.60079 to 14.44333, while increasing UIQI and CC to 0.866672 and 0.954173, respectively. These enhancements demonstrate that the Gaussian-process-guided search mechanism improves the optimizer's capacity to capture fine-grained vegetation textures, edge transitions, and small spectral fluctuations in complex green regions.

The *GF (Image 1)* and *GF (Image 2)* datasets show a similar trend, with GPbasedGO achieving the best overall performance. In particular, for GF (Image 2), the proposed technique consistently achieves the lowest distortion-related metrics and the highest similarity-related metrics among all analyzed algorithms,

indicating strong robustness under a variety of sensor and vegetation distributions. This higher performance indicates that GP-basedGO may efficiently balance spatial detail augmentation and spectral fidelity preservation in heavily textured vegetation images.

Fig. 10 shows visual comparisons that confirm the quantitative conclusions. GP-basedGO fusion findings show crisper canopy borders, sharper leaf-cluster textures, and more natural color transitions in densely vegetated areas. Furthermore, the suggested method preserves local contrast in shaded regions and complicated background boundaries, thereby avoiding the texture blurring and spectral inconsistencies seen in traditional optimization-based fusion methods. Visual observations are highly consistent with superior objective scores in Table 9, thus demonstrating the usefulness and robustness of GP-basedGO for remote sensing image fusion in green vegetation contexts.

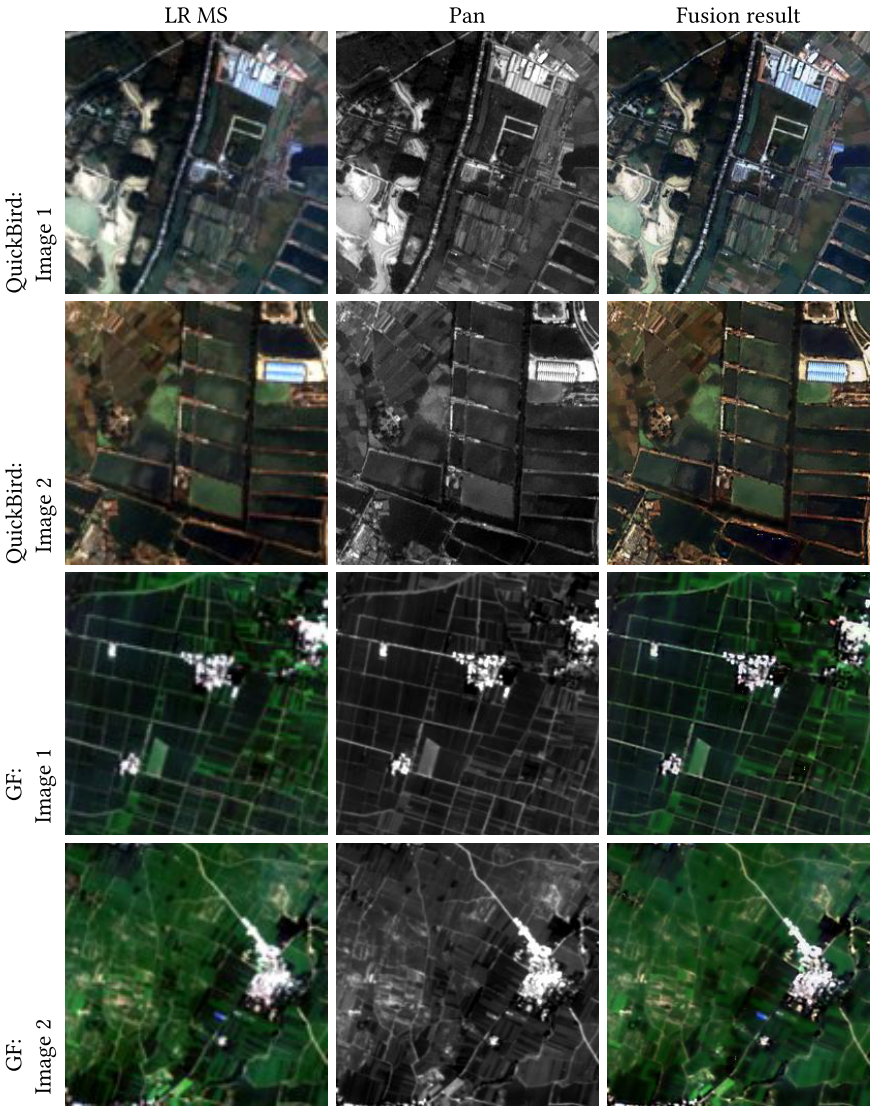


Figure 10: Visual assessment for marked region in green vegetation dataset.

4.2.3 Experiments in Water Scenarios

The quantitative results summarized in Table 10 further demonstrate the superior fusion capability of the proposed GPbasedGO in water scenarios. Across the four test cases, GPbasedGO consistently achieves the best or highly competitive performance, confirming its robustness in handling homogeneous regions and subtle spectral variations that are particularly challenging in water-dominated scenes.

Table 10: Experimental results of sixteen algorithms for urban scenarios.

Satellite	Algorithm	ERGAS	SAM	RASE	RMSE	UIQI	CC
QuickBird (Image1)	GPbasedGO	170.909232	9.023280233	35.81054868	17.83914455	0.928565367	0.887037769
	GO	**170.817759**	9.252257389	35.94462989	17.90593755	0.927700141	0.88501741
	PSO	170.9653093	9.390245494	36.25779246	18.06194053	0.92630525	0.88158404
	GA	170.9268924	9.271583808	36.01679255	17.94188562	0.927317636	0.884248497
	ASO	170.923375	9.265867962	36.08688789	17.97680385	0.92693241	0.883551367
	BOA	170.9144897	9.210929525	36.10090889	17.98378846	0.927536013	0.885198312
	SA	170.8372917	9.538861524	36.25511675	18.06060762	0.926549209	0.882011378
	CS	170.9625738	9.026135342	35.82788381	17.8477801	0.928325761	0.886414576
	GWO	170.9321328	9.498567187	36.31780629	18.09183662	0.92609248	0.880764651
	DE	170.8572623	9.14660537	35.88514908	17.87630698	0.927867721	0.885702297
	CRO	170.9870074	9.330768009	36.06689379	17.96684372	0.927232691	0.883999259
	CHO	170.9831258	9.477267664	36.34738777	18.10657273	0.92588291	0.880406083
	EMWPA	170.9413896	9.235163891	35.93662827	17.90195152	0.927812437	0.885446931
	FNO	170.91249	9.371525496	36.26003335	18.06305684	0.926195782	0.881536793
	LEE	170.9589365	9.385920783	36.08032888	17.97353645	0.927248966	0.883978252
	WOO	170.9390385	9.109313424	35.81273021	17.84023128	0.928291481	0.886614739
QuickBird (Image2)	GPbasedGO	33.05067904	8.161557414	35.27386592	18.22884103	0.93133415	0.958686662
	GO	33.07380461	8.216120919	35.3362686	18.26108952	0.931074107	0.958275523
	PSO	33.40663721	8.247896872	35.75310612	18.47650296	0.929414174	0.956043221
	GA	33.18583601	8.337988489	35.53353905	18.36303501	0.930374381	0.956993801
	ASO	33.06963311	8.238590049	35.47977162	18.33524906	0.930404543	0.956281129
	BOA	33.1695435	8.238087311	35.6105326	18.40282376	0.929892666	0.955801503
	SA	33.09226187	8.16800338	35.32258462	18.2540179	0.931147749	0.958477774
	CS	33.05465559	8.286881027	35.3587353	18.27269986	0.931060338	0.957814815
	GWO	33.07510905	8.187101282	35.32576453	18.25566122	0.931056815	0.958419432
	DE	33.28880168	8.176679642	35.48650757	18.33873007	0.930561266	0.958313838
	CRO	33.07587783	8.182594318	35.40769631	18.29800195	0.930737086	0.957392755
	CHO	33.13323568	8.204572551	35.51776902	18.35488537	0.930306352	0.956728529
	EMWPA	33.06367227	8.175305603	35.33002242	18.25786161	0.931090724	0.958131559
	FNO	33.08083094	8.228874637	35.48832033	18.33966686	0.930370804	0.956304998
	LEE	33.18223427	8.279033903	35.70836916	18.45338378	0.929490242	0.954722145
	WOO	33.0603242	8.23428585	35.60917226	18.40212076	0.929894571	0.954648994
GF (Image1)	GPbasedGO	72.34600016	4.756922011	34.75489301	10.4662868	0.968429276	0.988916738
	GO	72.34695348	4.771333925	34.8175049	10.4851421	0.968284622	0.988765011
	PSO	72.39474424	4.838388188	34.78157096	10.47432075	0.968372731	0.988898748
	GA	72.34678579	4.763440934	34.81453457	10.4842476	0.968309853	0.988748083
	ASO	72.34675447	4.87232769	34.77952997	10.47370611	0.968382339	0.988868569
	BOA	72.34684628	4.759973805	34.78284215	10.47470356	0.968386462	0.988853842
	SA	72.34853537	5.051688574	34.85743141	10.4971658	0.968214167	0.988643352
	CS	72.3475789	5.022212053	34.83049744	10.48905475	0.968268047	0.988720084
	GWO	72.35129389	5.140572139	34.86947301	10.50079207	0.968201758	0.988610245
	DE	72.34615392	4.810007363	34.7600229	10.46783164	0.968418372	0.988903243

(Continued)

Table 10 (continued)

Satellite	Algorithm	ERGAS	SAM	RASE	RMSE	UIQI	CC
	CRO	72.34874769	5.110420734	34.92008048	10.51603229	0.968135892	0.988479709
	CHO	72.34832911	5.119312041	34.89046776	10.50711455	0.96817585	0.988565966
	EMWPA	72.34890538	5.19476306	34.93456688	10.52039481	0.968098524	0.988445232
	FNO	72.34996253	5.042070116	34.86810129	10.50037899	0.968240168	0.988614611
	LEE	72.34768	4.873238011	34.84294542	10.49280341	0.968252634	0.988704089
	WOO	72.34835655	4.774666484	34.83635025	10.4908173	0.968248032	0.988731135
GF (Image2)	GPbasedGO	24.10013943	2.917317178	25.64187158	10.93238609	0.973131087	0.986797163
	GO	24.10878387	2.958591255	25.68181886	10.94941757	0.973039943	0.986585772
	PSO	24.11430906	3.003424013	25.69703111	10.9559033	0.973021281	0.986573863
	GA	24.11181716	3.04027284	25.68988491	10.95285652	0.973027825	0.986555967
	ASO	24.10671626	2.937040108	25.66356198	10.94163377	0.973080804	0.986697452
	BOA	24.11419311	2.939857606	25.66423357	10.9419201	0.973080911	0.986745275
	SA	24.11516331	2.994584725	25.68739296	10.95179408	0.973028542	0.986582077
	CS	24.10358296	3.0679063	25.69587488	10.95541034	0.973018509	0.986480801
	GWO	24.1076576	3.012066106	25.67335694	10.94580984	0.97306828	0.986650463
	DE	24.13474301	2.94592314	25.71060584	10.96169087	0.972977206	0.986564179
	CRO	24.12097982	3.044747793	25.69618524	10.95554266	0.973015816	0.986572656
	CHO	24.16744707	2.927462439	25.73594206	10.97249294	0.972936224	0.986629104
	EMWPA	24.11492692	2.981593977	25.67506437	10.9465378	0.973066985	0.986681082
	FNO	24.10501633	3.022960396	25.68190969	10.94945629	0.973050146	0.986594939
	LEE	24.10334538	**2.911747873**	25.6521901	10.93678538	0.973119663	0.98674451
	WOO	24.10283813	3.002126646	25.68249746	10.94970689	0.973056149	0.986586143

For *QuickBird (Image 1)*, GPbasedGO achieves the best values in SAM, RASE, RMSE, UIQI, and CC, while its ERGAS value remains highly competitive and only marginally inferior to GO. In particular, the proposed method significantly improves SAM to 9.0233 and CC to 0.8870, indicating that GPbasedGO is highly effective in preserving spectral smoothness and boundary consistency in water-land transition regions.

In the *QuickBird (Image 2)* scene, GPbasedGO delivers the optimal performance across all six evaluation metrics, clearly outperforming all competing methods. Compared with the original GO, the proposed method reduces ERGAS from 33.0738 to 33.0507 and improves CC from 0.9583 to 0.9587. These improvements, although numerically moderate, are highly meaningful in water scenarios where local textures are relatively weak and spectral distortions are more likely to dominate the fusion quality. This confirms that the Gaussian-process-guided search mechanism can more effectively capture subtle intensity gradients and preserve fine shoreline structures.

A similar superiority can be observed in the *GF (Image 1)* and *GF (Image 2)* datasets. In GF (Image 1), GPbasedGO achieves the best performance on all six metrics, demonstrating excellent detail preservation in smooth water surfaces and surrounding land boundaries. For GF (Image 2), the proposed method again attains the best values in ERGAS, RASE, RMSE, UIQI, and CC, while its SAM remains highly competitive and only slightly inferior to LEE. These results highlight the strong generalization capability of GPbasedGO under different sensor characteristics and diverse water distributions.

The visual comparisons in Fig. 11 further support the quantitative analysis. The fusion results generated by GPbasedGO exhibit clearer shoreline contours, more natural tonal transitions over water surfaces, and better preservation of weak boundary structures in shadowed or reflective regions. Compared with the

baseline methods, the proposed approach effectively suppresses spectral distortion and avoids the over-smoothing artifacts that commonly occur in homogeneous water areas. The high visual consistency between water bodies and adjacent land regions is in strong agreement with the superior UIQI and CC values reported in Table 10, further validating the effectiveness of GPbasedGO for remote sensing image fusion in water scenarios.

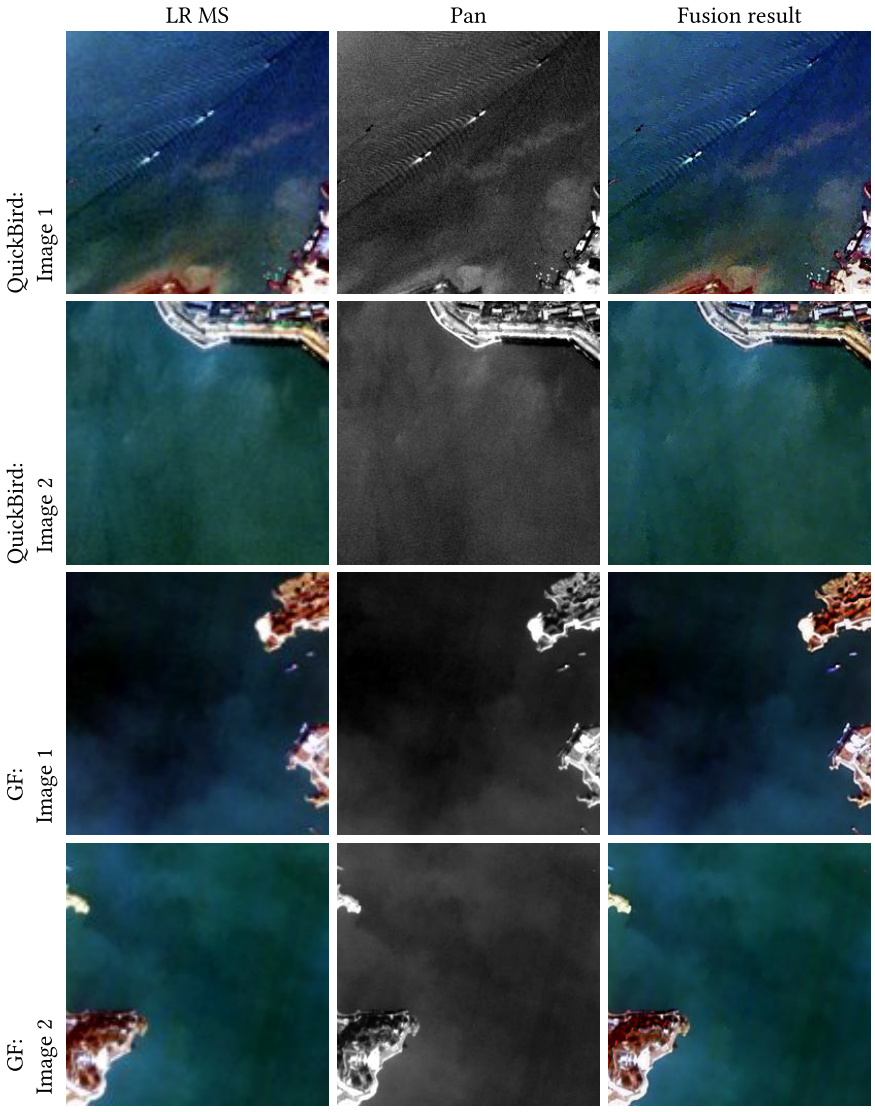


Figure 11: Visual assessment for marked region in water scenario dataset.

4.2.4 Experiments in Coastal Scenarios

Table 11 shows the good fusion performance of the proposed GP-based GO in coastal environments. GPbasedGO consistently outperforms or is highly competitive in all four test cases, demonstrating its strong ability to handle complex shoreline structures, mixed land-water boundaries, and significant spectral heterogeneity, all of which are common in coastal remote sensing scenes. These factors make coastal image fusion far more difficult than in homogeneous inland areas.

Table 11: Experimental results of sixteen algorithms for urban scenarios.

Satellite	Algorithm	ERGAS	SAM	RASE	RMSE	UIQI	CC
QuickBird (Image1)	GPbasedGO	6.092898	8.71684	26.92863	25.08116	0.954367	0.992877
	GO	6.126346	8.799241	27.05344	25.1974	0.953994	0.992074
	PSO	6.100574	8.745824	26.96054	25.11087	0.95425	0.992804
	GA	6.107492	8.753932	26.97943	25.12847	0.954247	0.992415
	ASO	6.101144	8.783679	26.95945	25.10986	0.954293	0.992588
	BOA	6.098438	8.752021	26.94748	25.09871	0.954329	0.992692
	SA	6.101153	8.768794	26.96198	25.11222	0.95429	0.992611
	CS	6.105964	8.781266	26.9852	25.13384	0.95418	0.992603
	GWO	6.102695	8.763028	26.96778	25.11762	0.954239	0.99259
	DE	6.125035	8.724937	27.06222	25.20558	0.954005	0.992722
	CRO	6.114237	8.781005	27.01131	25.15816	0.954112	0.992503
	CHO	6.102524	8.762552	26.96094	25.11125	0.954294	0.992572
	EMWPA	6.113133	8.749948	27.00539	25.15265	0.954175	0.992449
	FNO	6.107843	8.760913	26.97768	25.12684	0.954256	0.992446
	LEE	6.101687	8.773958	26.96136	25.11164	0.95429	0.992567
	WOO	6.100015	8.758428	26.95644	25.10705	0.954308	0.992643
QuickBird (Image2)	GPbasedGO	97.00118	6.353935	31.89221	15.26325	0.944103	0.938331
	GO	97.16706	6.361254	36.38002	17.41107	0.928916	0.914426
	PSO	102.743	6.352387	35.67517	17.07374	0.931426	0.917657
	GA	97.09284	6.479789	37.70919	18.0472	0.923946	0.905886
	ASO	97.07435	6.351026	36.93485	17.67661	0.927111	0.911022
	BOA	97.29196	6.38934	37.58737	17.98889	0.924468	0.90677
	SA	96.98097	6.431239	34.02502	16.28399	0.937353	0.926219
	CS	97.45984	6.362558	37.7199	18.05232	0.923941	0.904607
	GWO	97.07356	6.465045	40.64053	19.4501	0.91297	0.888214
	DE	97.08914	6.219977	37.56729	17.97929	0.92482	0.904553
	CRO	97.07195	6.361771	35.14681	16.82087	0.933349	0.921294
	CHO	97.32229	6.370312	35.27485	16.88215	0.93305	0.920023
	EMWPA	97.3989	6.329262	33.59799	16.07962	0.938515	0.928113
	FNO	97.06065	6.416857	33.98229	16.26354	0.937516	0.924987
	LEE	97.09113	6.302597	34.12387	16.3313	0.936723	0.925253
	WOO	97.22545	6.42271	32.94513	15.76717	0.940923	0.931721
GF (Image1)	GPbasedGO	46.58689	3.260379	20.70639	14.02412	0.97538	0.990128
	GO	46.58923	3.384585	20.8333	14.11008	0.975083	0.989603
	PSO	46.59337	3.287801	20.73864	14.04596	0.975304	0.98999
	GA	46.58976	3.334288	20.92214	14.17024	0.974872	0.989085
	ASO	46.58778	3.335589	20.71974	14.03316	0.97535	0.990095
	BOA	46.59363	3.370919	20.72207	14.03474	0.975345	0.990093
	SA	46.58779	3.362571	20.74918	14.0531	0.975283	0.989975
	CS	46.58945	3.347695	20.8469	14.11928	0.975051	0.989525
	GWO	46.58749	3.368913	20.71009	14.02663	0.975374	0.990118
	DE	46.5892	3.339077	20.90383	14.15785	0.974915	0.989182
	CRO	46.58964	3.339122	20.78292	14.07595	0.975201	0.989828
	CHO	46.59025	3.314075	20.87325	14.13713	0.974987	0.989348
	EMWPA	46.58726	3.284912	20.72696	14.03805	0.975331	0.990033
	FNO	46.58878	3.425831	20.74893	14.05293	0.975286	0.989987
	LEE	46.5873	3.272577	20.76726	14.06534	0.975237	0.989843
	WOO	46.58824	3.330665	20.78794	14.07935	0.975189	0.989771

(Continued)

Table 11 (continued)

Satellite	Algorithm	ERGAS	SAM	RASE	RMSE	UIQI	CC
GF (Image2)	GPbasedGO	50.60425	2.838725	25.70182	14.38068	0.967407	0.984249
	GO	50.90271	2.852793	31.4766	17.61179	0.951874	0.966127
	PSO	51.82908	2.851152	26.82466	15.00894	0.96459	0.982181
	GA	50.98863	2.897747	27.05792	15.13945	0.963996	0.979104
	ASO	50.87831	2.878466	29.11019	16.28773	0.958515	0.970787
	BOA	50.85513	2.874673	26.86369	15.03077	0.964454	0.979494
	SA	50.66889	2.873479	26.63386	14.90218	0.965044	0.979505
	CS	50.84477	2.877819	28.66768	16.04014	0.959699	0.971717
	GWO	50.75855	2.850832	27.85621	15.58611	0.961923	0.977761
	DE	50.85738	2.949287	31.31815	17.52313	0.952349	0.964054
	CRO	50.80827	2.84784	26.11738	14.6132	0.966382	0.983222
	CHO	51.26257	2.862301	27.14928	15.19057	0.96374	0.980303
	EMWPA	50.7572	2.854122	26.90096	15.05162	0.964393	0.980757
	FNO	50.92287	2.875396	28.01932	15.67737	0.961446	0.974739
	LEE	50.78041	2.85356	29.43866	16.47152	0.957621	0.971536
	WOO	51.1908	2.90665	27.75945	15.53197	0.962176	0.976924

In *QuickBird (Image 1)*, GPbasedGO outperforms all six evaluation metrics: ERGAS (6.092898), SAM (8.71684), RASE (26.92863), RMSE (25.08116), UIQI (0.954367), and CC (0.992877). Compared to the original GO, the suggested method greatly improves both spectral and structural consistency, particularly in UIQI and CC, indicating that GP-basedGO is more effective at retaining fine shoreline textures and decreasing spectral distortion in coastal transition zones.

In the tough QuickBird (Image 2) environment, GPbasedGO achieves the best values in RASE, RMSE, UIQI, and CC, while its ERGAS and SAM remain highly competitive. Although SA and DE perform marginally better in ERGAS and SAM, respectively, GP-basedGO performs much better overall in similarity-related metrics. In particular, as compared to GO, the suggested method increases UIQI from 0.928916 to 0.944103 and CC from 0.914426 to 0.938331, demonstrating its improved capacity to maintain structural continuity and edge integrity in extremely varied coastal regions.

GF (Image 1) and GF (Image 2) datasets show similar advantages. For GF (Image 1), GP-basedGO outperforms all six assessment parameters, demonstrating exceptional retention of fine coastline details and spectral fidelity. In GF (Image 2), the suggested technique achieves the best performance in all six metrics, with especially considerable gains in ERGAS, RASE, RMSE, UIQI, and CC when compared to competing algorithms. These findings demonstrate that the Gaussian-process-guided search mechanism can significantly improve optimization accuracy and strike a better balance between spatial detail augmentation and spectrum information preservation across a variety of sensor attributes and coastal distributions.

The visual comparisons in Fig. 12 support the quantitative results. The fusion images created by GPbasedGO show crisper shoreline contours, finer border transitions between land and water, and more natural tonal fluctuations in shallow-water and reflective areas. Compared to baseline techniques, the suggested methodology effectively reduces spectral distortion and prevents boundary blurring, which is typical in coastal fusion tasks. Furthermore, the retention of faint texture information in ports, tidal flats, and coastline vegetation regions has been improved. These visual observations align with the superior objective results given in Table 11, highlighting the robustness and usefulness of GP-basedGO for remote sensing image fusion in coastal environments.

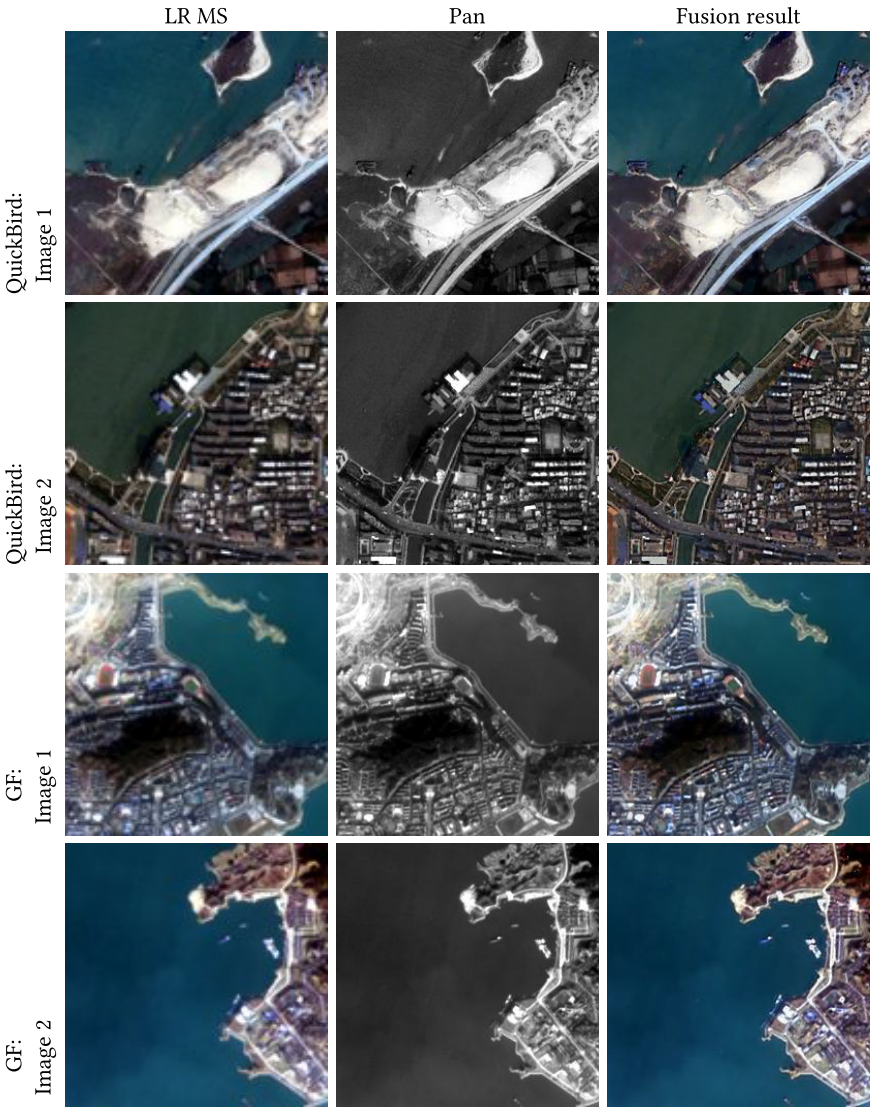


Figure 12: Visual assessment for marked region in coast dataset.

5 Conclusions

In this paper, a Genetic Programming-based Metaheuristic framework (GP-MAs) is developed to address the constraints of standard metaheuristic algorithms imposed by hand-crafted updating strategies. The suggested system, which incorporates Genetic Programming (GP) into the search process, allows for the automatic evolution of symbolic update equations, boosting the adaptability and flexibility of metaheuristic optimization. Based on this architecture, a hybrid optimizer called GPbasedGO is created, with the Growth Optimizer (GO) as the baseline algorithm. Extensive studies on the CEC2022 benchmark suite show that GP-basedGO has competitive convergence behavior, optimization performance, and stability when compared to many representative state-of-the-art algorithms. Furthermore, the efficacy of GP-basedGO is assessed in multispectral and panchromatic image fusion tasks across urban, vegetation, water, and coastal situations, with the suggested approach outperforming many image quality measures such as ERGAS, SAM, RMSE, UIQI, and CC. The experimental results show that the proposed GP-MAs framework offers a flexible and partially automated approach to symbolic strategy evolution in metaheuristic optimization. This study

validates GP-MAs with GO as a representative human-inspired optimizer, proving the viability of GP-assisted update rule evolution for representative optimization problems. However, the current study does not fully cover highly dynamic, limited, or non-stationary optimization situations, and further validation of swarm-based and physics-based optimizers remains a future task. Future research will expand the suggested framework to multi-objective, large-scale, and discrete optimization issues, as well as investigate its integration with deep learning and reinforcement learning approaches.

Although the proposed GP-MAs framework achieves competitive optimization performance and demonstrates its effectiveness in remote sensing image fusion, some limitations remain. The theoretical convergence analysis of the evolved symbolic update rules has not been fully established, and the computational cost of the offline GP evolution process requires further quantitative investigation. Future research will focus on developing more rigorous theoretical analysis methods, improving the efficiency of the GP evolution process, and extending the proposed framework to a wider range of metaheuristic algorithms and complex optimization scenarios.

Acknowledgement: None.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Shu-Chuan Chu and Hongmei Yang; methodology, Jeng-Shyang Pan, Wenda Li, and Hongmei Yang; software, Wenda Li; validation, Jeng-Shyang Pan and Wenda Li; formal analysis, Wenda Li, Zhi-Gang Du, and Lingping Kong; investigation, Wenda Li; resources, Shu-Chuan Chu; data curation, Wenda Li and Shu-Chuan Chu; writing—original draft preparation, Wenda Li; writing—review and editing, Shu-Chuan Chu; visualization, Wenda Li; supervision, Jeng-Shyang Pan, Zhi-Gang Du, Hongmei Yang, and Lingping Kong; project administration, Jeng-Shyang Pan; funding acquisition, Not applicable. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are openly available in GitHlub at https://github.com/liwenmda/Metaheuristic-Algorithms.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Dokeroglu T, Sevinc E, Kucukyilmaz T, Cosar A. A survey on new generation metaheuristic algorithms. Comput Ind Eng. 2019;137(5):106040. doi:10.1016/j.cie.2019.106040.
2. Rajwar K, Deep K, Das S. An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges. Artif Intell Rev. 2023;56(11):13187–257. doi:10.1007/s10462-023-10470-y.
3. Nguyen TT, Pan JS, Dao TK. An improved flower pollination algorithm for optimizing layouts of nodes in wireless sensor network. IEEE Access. 2019;7:75985–98. doi:10.1109/access.2019.2921721.
4. Du ZG, Pan JS, Chu SC, Luo HJ, Hu P. Quasi-affine transformation evolutionary algorithm with communication schemes for application of RSSI in wireless sensor networks. IEEE Access. 2020;8:8583–94. doi:10.1109/access.2020.2964783.
5. Zhou G, Xie Y, Lan H, Tian W, Buyya R, Wu K. Information interaction and partial growth-based multi-population growable genetic algorithm for multi-dimensional resources utilization optimization of cloud computing. Swarm Evol Comput. 2024;87(4):101575. doi:10.1016/j.swevo.2024.101575.
6. Abasi AK, Aloqaily M, Guizani M, Ouni B. Metaheuristic algorithms for 6G wireless communications: recent advances and applications. Ad Hoc Netw. 2024;158(4):103474. doi:10.1016/j.adhoc.2024.103474.

7. Wang L, Pan Z, Wang J. A review of reinforcement learning based intelligent optimization for manufacturing scheduling. Complex Syst Model Simul. 2021;1(4):257–70. doi:10.23919/csms.2021.0027.
8. Talbi EG. Machine learning into metaheuristics: a survey and taxonomy. ACM Comput Surv. 2021;54(6):1–32. doi:10.1145/3459664.
9. Wang X, Wu Y. An improved deep spiking neural network model based on bionic optimization algorithm. J Netw Intell. 2024;9(1):160–75. doi:10.1109/icaci.2012.6463299.
10. Elsayed HA, Hameed MF, El Sherbiny MM. Image classification using decision tree classifier and features extraction using Hough transform and genetic algorithm. J Inf Hiding Multimed Signal Process. 2025;16(1):438–52.
11. Mitchell M. An introduction to genetic algorithms. Cambridge, MA, USA: MIT Press; 1998.
12. Pan JS, Chen J, Li J, Meng Z. Differential evolution with exponential crossover: a survey. Comput Sci Rev. 2026;62(4):100990. doi:10.1016/j.cosrev.2026.100990.
13. Beyer HG, Schwefel HP. Evolution strategies—a comprehensive introduction. Nat Comput. 2002;1:3–52.
14. Simon D. Biogeography-based optimization. IEEE Trans Evol Comput. 2008;12(6):702–13. doi:10.1109/tevc.2008.919004.
15. Shams B. Using network properties to evaluate targeted immunization algorithms. Netw Biol. 2014;4(3):74.
16. Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of ICNN'95-International Conference on Neural Networks; 1995 Nov 27–Dec 1; Perth, WA, Australia. p. 1942–8.
17. Blum C. Ant colony optimization: introduction and recent trends. Phys Life Rev. 2005;2(4):353–73.
18. Karaboga D, Akay B. A comparative study of artificial bee colony algorithm. Appl Math Comput. 2009;214(1):108–32. doi:10.1016/j.amc.2009.03.090.
19. Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Softw. 2014;69:46–61. doi:10.1016/j.advengsoft.2013.12.007.
20. Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw. 2016;95(12):51–67. doi:10.1016/j.advengsoft.2016.01.008.
21. Abdollahzadeh B, Gharehchopogh FS, Khodadadi N, Mirjalili S. Mountain gazelle optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems. Adv Eng Softw. 2022;174(3):103282. doi:10.1016/j.advengsoft.2022.103282.
22. Bertsimas D, Tsitsiklis J. Simulated annealing. Stat Sci. 1993;8(1):10–5. doi:10.1214/ss/1177011077.
23. Rashedi E, Nezamabadi-Pour H, Saryazdi S. GSA: a gravitational search algorithm. Inf Sci. 2009;179(13):2232–48. doi:10.1016/j.ins.2009.03.004.
24. Mirjalili S. SCA: a sine cosine algorithm for solving optimization problems. Knowl-based Syst. 2016;96(63):120–33. doi:10.1016/j.knosys.2015.12.022.
25. Mirjalili S, Mirjalili SM, Hatamlou A. Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl. 2016;27(2):495–513. doi:10.1007/s00521-015-1870-7.
26. Rao RV, Savsani VJ, Vakharia DP. Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput-Aided Des. 2011;43(3):303–15. doi:10.1016/j.cad.2010.12.015.
27. Geem ZW, Kim JH, Loganathan GV. A new heuristic optimization algorithm: harmony search. Simulation. 2001;76(2):60–8. doi:10.1177/003754970107600201.
28. Satapathy S, Naik A. Social group optimization (SGO): a new population evolutionary optimization technique. Complex & Intell Syst. 2016;2(3):173–203. doi:10.1007/s40747-016-0022-8.
29. Zhang W, Pan K, Li S, Wang Y. Special forces algorithm: a novel meta-heuristic method for global optimization. Math Comput Simul. 2023;213:394–417.
30. Pan JS, Zhang LG, Wang RB, Snášel V, Chu SC. Gannet optimization algorithm: a new metaheuristic algorithm for solving engineering optimization problems. Math Comput Simul. 2022;202:343–73.
31. Wang J, Chen Y, Lu C, Heidari AA, Wu Z, Chen H. The status-based optimization: algorithm and comprehensive performance analysis. Neurocomputing. 2025;647:130603.
32. Liang Z, Wang Z, Mohamed AW. A novel hybrid algorithm based on improved marine predators algorithm and equilibrium optimizer for parameter extraction of solar photovoltaic models. Heliyon. 2024;10(19):e38412. doi:10.1016/j.heliyon.2024.e38412.

33. Song H, Bei J, Zhang H, Wang J, Zhang P. Hybrid algorithm of differential evolution and flower pollination for global optimization problems. Expert Syst Appl. 2024;237(14):121402. doi:10.1016/j.eswa.2023.121402.
34. Zhao J, Chen D, Xiao R, Chen J, Pan JS, Cui ZH, et al. Multi-objective firefly algorithm with adaptive region division. Appl Soft Comput. 2023;147(7):110796. doi:10.1016/j.asoc.2023.110796.
35. Ma Z, Guo H, Chen J, Li Z, Peng G, Gong YJ, et al. Metabox: a benchmark platform for meta-black-box optimization with reinforcement learning. Adv Neural Inf Process Syst. 2023;36:10775–95. doi:10.5555/3666122.3666596.
36. Finn C, Abbeel P, Levine S. Model-agnostic meta-learning for fast adaptation of deep networks. In: Proceedings of the 34th International Conference on Machine Learning; 2017 Aug 6–11; Sydney, NSW, Australia. p. 1126–35.
37. Liu F, Tong X, Yuan M, Lin X, Luo F, Wang Z, et al. Evolution of heuristics: towards efficient automatic algorithm design using large language model. arXiv:2401.02051. 2024.
38. Drugan MM. Reinforcement learning versus evolutionary computation: a survey on hybrid algorithms. Swarm Evol Comput. 2019;44:228–46.
39. Salimans T, Ho J, Chen X, Sidor S, Sutskever I. Evolution strategies as a scalable alternative to reinforcement learning. arXiv:1703.03864. 2017.
40. Guo H, Ma Y, Ma Z, Chen J, Zhang X, Cao Z, et al. Deep reinforcement learning for dynamic algorithm selection: a proof-of-principle study on differential evolution. IEEE Trans Syst Man Cybern Syst. 2024;54(7):4247–59.
41. Ma Z, Guo H, Chen J, Peng G, Cao Z, Ma Y, et al. LLaMoCo: instruction tuning of large language models for optimization code generation. arXiv:2403.01131. 2024.
42. Gomes HS, Léger B, Gagné C. Meta learning black-box population-based optimizers. arXiv:2103.03526. 2021.
43. AhmadiTeshnizi A, Gao W, Udell M. OptiMUS: scalable optimization modeling with (MI) LP solvers and large language models. arXiv:2402.10172. 2024.
44. Wang J, Zheng Y, Zhang Z, Peng H, Wang H. A novel multi-state reinforcement learning-based multi-objective evolutionary algorithm. Inf Sci. 2025;688(9):121397. doi:10.1016/j.ins.2024.121397.
45. Ma Z, Guo H, Gong YJ, Zhang J, Tan KC. Toward automated algorithm design: a survey and practical guide to meta-black-box-optimization. IEEE Trans Evol Comput. 2025;30(2):667–87.
46. Guo H, Ma S, Huang Z, Hu Y, Ma Z, Zhang X, et al. Reinforcement learning-based self-adaptive differential evolution through automated landscape feature learning. In: Proceedings of the Genetic and Evolutionary Computation Conference; 2025 Jul 14–18; Malaga, Spain. p. 1117–26.
47. Koza JR. Genetic programming as a means for programming computers by natural selection. Stat Comput. 1994;4(2):87–112. doi:10.1007/bf00175355.
48. Xue X, Lin JCW. Heterogeneous sensor integration through co-evolutionary genetic programming with multiple individual representations. Swarm Evol Comput. 2025;98(5):102098. doi:10.1016/j.swevo.2025.102098.
49. Zhang Q, Gao H, Zhan ZH, Li J, Zhang H. Growth optimizer: a powerful metaheuristic algorithm for solving continuous and discrete global optimization problems. Knowl Based Syst. 2023;261:110206.
50. Peng Y, Gu S, Liang Y, Ouyang K, Li Y, Wang K, et al. Wave optics optimizer: a novel meta-heuristic algorithm for engineering optimization. Commun Nonlinear Sci Numer Simul. 2026;152:109337.
51. Abdullah M, Alimgeer KS, Hafeez G, Alghamdi B, Alsafran AS, Babar MZ. Electromagnetic wave propagation algorithm: a novel electromagnetic wave propagation-inspired optimizer for engineering applications. Ain Shams Eng J. 2025;16(11):103615.
52. Taheri A, RahimiZadeh K, Baumbach J, Beheshti A, Zolotareva O, Al-Betar MA, et al. Farthest better or nearest worse optimizer: a novel metaheuristic algorithm. Artif Intell Rev. 2025;59(2):79. doi:10.1007/s10462-025-11443-z.
53. Sharifi T, Mirsalim M, Soleimanian Gharehchopogh F, Mirjalili S. Cultural history optimization algorithm: a new human-inspired metaheuristic algorithm for engineering optimization problems. Neural Comput Appl. 2025;37(25):21009–68. doi:10.1007/s00521-025-11379-z.
54. Chen H, Ahmadianfar I, Heidari AA, Kordani M, Samadi Koucheksaraee A, Liang G. LEE: a physics-inspired optimizer based on LangEvin equation. Neurocomputing. 2026;666:132288.