A Hybrid Fuzzy-IGA-APSO Framework for Real-Time Urban Landscape Optimization in Smart Cities

Yang Zhang
Zhengzhou Academy of Fine Arts, Zhengzhou, 451450, China
zhang_yangy@outlook.com

Keywords: smart city, hybrid soft computing, dynamic landscape generation, fuzzy logic, genetic algorithm, particle swarm optimisation, real-time decision making, energy efficiency optimisation

Received: July 3, 2025

With the rapid development of smart cities, efficient and real-time urban landscape management has become an urgent research topic. This paper proposes a Hybrid Soft Computing Framework (HSCF) that combines Fuzzy Logic, Improved Genetic Algorithm (IGA), and Adaptive Particle Swarm Optimization (APSO) to dynamically optimize urban systems such as lighting and irrigation. By integrating heterogeneous sensor data (e.g., weather, pedestrian flow, and traffic conditions), the framework senses environmental changes and makes optimization decisions in real time.The Fuzzy Logic module handles low-latency adjustments, such as dynamically tuning lighting brightness based on crowd density, achieving response times of less than 100 ms. The IGA performs mid-term optimization of multi-objective landscape layouts (e.g., energy efficiency, aesthetics, and functionality) every 5 minutes, evolving Pareto-optimal solutions through non-dominated sorting and crowding distance analysis with a population size of 50, crossover rate of 0.8 – 0.95, and mutation rate of 0.05 – 0.15. The APSO continuously refines these solutions using real-time spatio-temporal data, adaptively balancing exploration and exploitation through inertia weight adjustments (ranging from 0.4 to 0.9) and acceleration constants (c_1 = 1.2 – 1.8, c_2 = 1.2 – 2.0).Experimental results demonstrate that HSCF outperforms traditional methods (e.g., FLC and PSO), achieving 16.2%-22.7% energy consumption reduction, 36.6% water savings in irrigation systems, and maintaining stability under extreme weather and ±20% data noise. Key innovations include dynamic spatio-temporal data fusion, real-time decision-making, and joint fitness evaluation across layers. Future work will focus on scalability and integration of additional data sources (e.g., UAV-derived 3D maps) to address more complex urban management tasks. This framework provides a replicable, data-driven solution for adaptive smart city landscape management.

Povzetek:

1 Introduction

With the rapid development of information technology and the continuous promotion of the smart city concept, urban management is facing unprecedented challenges. Smart city construction aims to optimise the allocation of urban resources, improve the efficiency of urban operation and enhance the quality of life of citizens through modern information technology. As an important part of the smart city, the dynamic adjustment and management of urban landscapes plays a crucial role in enhancing the quality of the urban environment, conserving resources, and promoting sustainable development. Traditional urban landscape management methods often rely on static rules and manual control, and cannot be flexibly adjusted according to the actual situation.

Scholars have explored the synergistic innovation path of data-driven and intelligent algorithms through multidisciplinary crossover. Zhu et al.[1] took the lead in constructing a dynamic urban planning framework integrating hybrid artificial intelligence and big data. Their proposed coupled model of deep reinforcement learning and multi-objective optimisation provides real-time decision support for the dynamic configuration of landscape elements, but has not yet solved the complexity of high-dimensional data feature interactions. To address this limitation, Yu et al.[2] developed a multi-objective landscape planning model based on the NSGA-II algorithm, which achieves a Pareto-optimal solution for landscape aesthetic value and ecological benefits by integrating geospatial big data and ecological constraints, but its static optimisation characteristics are difficult to adapt to the needs of dynamic urban evolution. In this regard, Khan et al.[3] innovatively proposed a multi-scale modelling framework to temporally and spatially correlate macro urban form and micro landscape elements through hierarchical reinforcement learning, and the cross-scale feedback mechanism they constructed significantly enhanced the adaptive ability of the landscape system. Wang and Ma[4] further introduced digital twin technology into this field, and developed a collaborative energy-landscape digital twin platform, which can be used for the development of the landscape system through real-time data and ecological constraints,

and the energy-landscape collaboration. digital twin platform, which realizes the dynamic adaptation of renewable energy facilities and landscape layout through a real-time data stream-driven particle swarm optimization algorithm, but the network latency problem affects the response speed of the system. Ruan et al.[5], on the other hand, expanded the research boundaries from the dimension of social computing, and used a fuzzy cognitive map and a spatio-temporal clustering algorithm to analyze social media data streams, revealing the dynamic mapping relationship between public behavioural patterns and landscape Bibri[6] systematically demonstrates the theoretical support of urban computing technology for dynamic landscape planning, and proposes to embed online learning mechanism into the planning decision-making cycle, which is a theoretical framework that effectively solves the synergy problem of long-term and short-term planning objectives in traditional methods. Kuru[7] pioneers the introduction of UAV swarm intelligence technology, and the use of UAVs in the planning process. introduced the UAV swarm intelligence technology to achieve aerial dynamic reorganisation of landscape elements through a distributed swarm algorithm, and its proposed 3D spatial optimisation model breaks through the limitations of traditional 2D planning. Notably, the hybrid network security architecture constructed by Sengan et al.[8] provides an important guarantee for the data security of the dynamic landscape system, and the quantum encrypted data stream transmission protocol designed by them effectively guards against the risk of cyber-attacks during the real-time optimisation process. There is still room for improvement in existing research in terms of algorithm fusion, real-time response speed and human-computer collaboration mechanism, and in the future, the in-depth fusion of digital twin, swarm intelligence and blockchain technology needs to be strengthened in order to construct a more resilient and adaptive smart landscape generation system.

Table 1: Comparison of related work

Researchers	Method	Application Domain	Performance Metrics	Limitations
Zhu et al.	Deep Reinforcement Learning + Multi-Objective Optimization	Lighting, Landscape Configuration	Energy reduction: 12%	Complexity of high-dimensional data interactions remains unresolved
Yu et al.	NSGA-II	Landscape Aesthetic Value and Ecological Benefits	Pareto-optimal solutions	Static optimization, difficult to adapt to dynamic urban evolution
Khan et al.	Hierarchical Reinforcement Learning	Macro Urban Form and Micro Landscape Elements	Cross-scale feedback enhances system adaptability	Lacks real-time data processing capabilities
Wang and Ma	Digital Twin + Particle Swarm Optimization	Renewable Energy Integration	Response time: 200 ms	Network latency affects system response speed
Ruan et al.	Fuzzy Cognitive Map + Spatio-Temporal Clustering Algorithm	Social Media Data Analysis	Reveals dynamic mapping between public behavior and landscape	Not directly applied to real-world landscape optimization
Bibri	Hybrid Urban Computing Framework	Dynamic Landscape Planning	Supports embedding of online learning mechanisms	Requires further validation in real-world scenarios
Kuru	UAV Swarm Intelligence	3D Spatial Optimization	Breaks through limitations of traditional 2D planning	High computational complexity, lacks real-time support

The aim of this paper is to propose a hybrid soft computing framework (HSCF) for real-time dynamic urban landscape optimisation, including the regulation of lighting and irrigation systems, by combining fuzzy logic, Improved Genetic Algorithm (IGA) and Adaptive Particle Swarm Optimisation (APSO). Logic, IGA and APSO, can outperform traditional methods (e.g. Fuzzy Logic Control and Particle Swarm Optimisation) in terms of real-time data processing and dynamic optimisation; the hybrid framework has significant advantages in terms of energy-efficiency optimisation, and the HSCF is able to achieve higher energy savings than the existing studies, while ensuring the dynamic adaptability and robustness of the landscape

system. The system boundaries are defined as follows: the algorithmic part will be tested in a simulation environment to verify its effectiveness in dynamic data processing and optimisation performance, while the practical application will be limited to lighting and irrigation systems in urban landscapes, with real-time data (e.g., weather, foot traffic, traffic conditions, etc.) obtained through IoT sensors; for experimental conditions and reproducibility, all experiments are conducted under the same hardware and software conditions to ensure the reproducibility of the results, the experimental data include the data collected from real scenarios as well as the simulation-generated data to cover different environmental conditions (e.g., extreme weather and emergent conditions), and the parameter settings (e.g., the population size of the IGA, the inertia weights of the APSO, etc.) are finely tuned based on the pre-experiments in order to balance the exploratory and developmental capabilities. The HSCF framework proposed in this paper has the following innovations: the combination of real-time and dynamic optimisation, where the framework achieves real-time adjustment through fuzzy logic, and mid-term optimisation and dynamic adaptation through IGA and APSO, which effectively solves the problem of insufficient real-time responsiveness and dynamic adaptability in the existing researches; multi-objective optimisation functionality, where the framework is able to optimise multiple objectives (e.g., energy-efficiency, aesthetics, and functionality) at the same time and achieve global adaptive capability through the Joint adaptation assessment mechanism to achieve global optimisation; Data-driven strategy, the framework adopts a spatio-temporal data fusion approach to unify information from different data sources into a spatio-temporally consistent framework to improve data processing efficiency and optimisation accuracy.

2 Hybrid soft computing framework design

2.1 System architecture

Figure 1 visualises the architecture of the hybrid soft computing framework, which consists of three main layers: the data layer, the processing layer and the output layer. The data layer is responsible for collecting and pre-processing data, where IoT sensors collect information such as weather data, people flow and traffic conditions. The data is preprocessed to align spatio-temporal data and filter noise. The processing layer focuses on core algorithms, including fuzzy logic controllers, Improved Genetic Algorithm (IGA) and Adaptive Particle Swarm Optimisation (APSO). These algorithms work together through a hybrid optimisation engine to dynamically tune the solution. The output layer uses the optimised solution to generate visualisations through platforms such as GIS and Unity3D, as well as issuing control commands for systems such as lighting and irrigation. The diagram illustrates the flow of data between these layers, from data collection and pre-processing to optimisation and control command issuance, where feedback loops ensure continuous adjustment of the system. This structure demonstrates the system's ability to handle real-time data processing, decision-making and control in a dynamic and collaborative manner, with each layer playing a specific role in the overall operation of the framework.

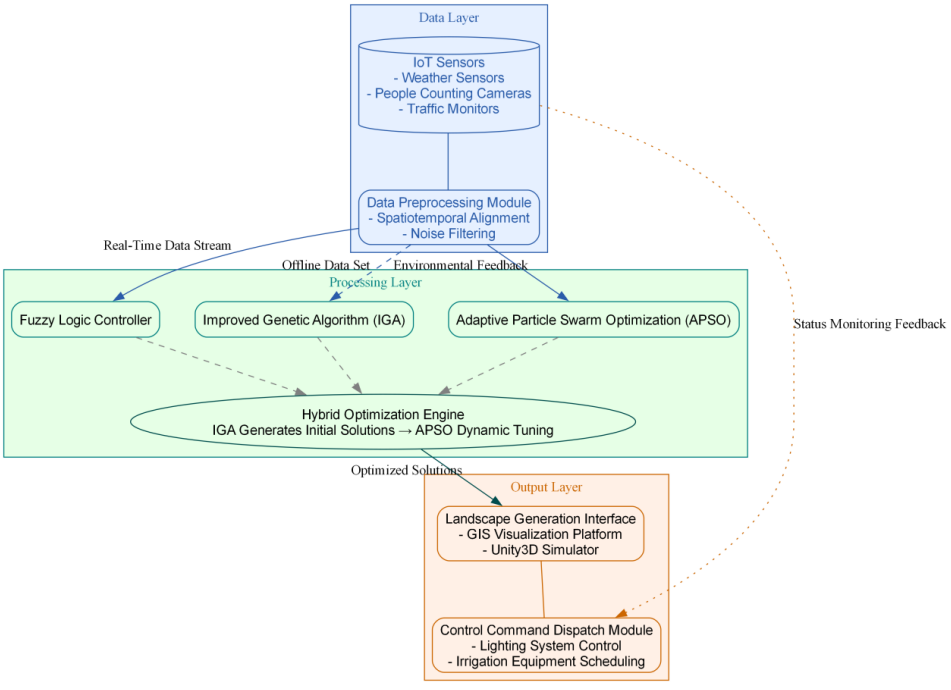

Figure 1: System architecture diagram

2.2 Core algorithm fusion mechanism

2.2.1 Fuzzy logic

First of all, it is necessary to define the affiliation function for the input variables. Here, we take 'crowd density' (D) and 'light brightness' (L) as examples to illustrate the definition of the affiliation function[9].

The affiliation function of the crowd density D: Assumptions $D \in [0,1]$, denotes the flow density from 'low' to 'high'.

We define the affiliation function of D as follows:

$$\mu_D(D) = \begin{cases} 1-D, & 0 \le D \le 0.5 \\ 2-D, & 0.5 < D \le 1 \end{cases} \quad (1)$$

Where $\mu_D(D)$ represents the degree of affiliation of the crowd density D, indicating the degree to which the value belongs to a 'low' or 'high' crowd density.

The affiliation function of light brightness L: Assume that $L \in [0,1]$, represents the light brightness from 'dark' to 'bright'. Define the affiliation function of L as:

$$\mu_L(L) = \begin{cases} 1-L, & 0 \le L \le 0.5 \\ 2-L, & 0.5 < L \le 1 \end{cases} \quad (2)$$

Where, $\mu_L(L)$ represents the light brightness L affiliation, reflecting the light brightness belongs to the degree of 'dark' or 'bright'.

Fuzzy rules produce fuzzy outputs based on fuzzy relationships between input variables[10-11]. Here, we assume that the control objective is to adjust the light luminance L based on the crowd density D and construct the fuzzy rule set. For example:

Rule 1: If crowd density is high (D high), then light luminance is high (L high).

Rule 2: If the crowd density is low (D low), the light luminance is low (L low).

According to this rule, we can use the following fuzzy language to express:

High crowd density: D is 'high'.

Low crowd density: D is 'low'.

High brightness: L is 'high'.

Low brightness: L is 'low'.

The fuzzy inference process includes rule activation, fuzzy synthesis and defuzzification. The following are the details of the process:

(1) Rule activation

Based on the input variables D and the affiliation function, we activate the fuzzy rule using the 'minimum value method'. For example, if the crowd density D = 0.7 (higher crowd density) and the light luminance L = 0.3 (lower luminance), the rule activation value is calculated in the following way:

$$\text{Rule1Activationvalue}.\min\left(\mu_D(0.7), \mu_L(0.3)\right) = \min(0.3, 0.7) = 0.3$$

$$\text{Rule2Activation value}.\min\left(\mu_D(0.7), \mu_L(0.3)\right) = \min(0.3, 0.7) = 0.3$$

(2) Fuzzy Synthesis

Through rule activation, we can synthesise the outputs of multiple rules together. Assuming that there are multiple fuzzy rules producing results, we use the 'maximum value method' to synthesise the output of each rule. The result of each rule is multiplied by the activation value to produce the final fuzzy output[12]. Assume that the output of the rule is an affiliation function of the brightness of the light L. The output of the synthesised rule will be a function of the brightness of the light. The synthesised output has the following affiliation function:

$$\mu_{L_{\text{final}}}(L) = \max\left(\mu_{L_{\text{rule1}}}(L), \mu_{L_{\text{rule2}}}(L)\right) \quad (3)$$

where $\mu_{L_{\text{final}}}(L)$ and $\mu_{L_{\text{rule2}}}(L)$ are outputs activated by rules.

(3) Defuzzification

The defuzzification process converts fuzzy values into actual outputs. The most commonly used defuzzification method is the 'centre of gravity method':

$$L_{\text{output}} = \frac{\int_0^1 L \cdot \mu_L(L)\, dL}{\int_0^1 \mu_L(L)\, dL} \quad (4)$$

Where L_{output} is the output light luminance value and $\mu_L(L)$ is the output affiliation function after fuzzy synthesis.

The centre of gravity method calculates the 'centre of gravity' of the output affiliation function to get the final light brightness value. Through this process, the fuzzy logic can adjust the light brightness in real time according to the fuzzy relationship between the crowd density and the current light brightness[13-14].

Assume that the landscape layout consists of n regions, and the configuration of each region is represented by $x_i = (x_{i1}, x_{i2}, \ldots, x_{im})$, where x_{ij} is the jth decision variable for the ith region. The objective function can be expressed as:

$$f(x) = w_1 E(x) + w_2 A(x) + w_3 F(x) \quad (5)$$

Among them: $E(x)$ Indicates energy consumption, reflecting the energy consumption required in each area of the layout.

$A(x)$ Indicates aesthetics and measures the visual impact of the layout.

$F(x)$ Indicates functionality and measures the efficiency and ease of use of the layout.

w_1, w_2, w_3 weighting factors for energy consumption, aesthetics and functionality, respectively.

Fuzzy logic and the multi-objective function f(x) serve distinct yet complementary roles in the Hybrid Soft Computing Framework (HSCF). Here's a concise explanation of their integration:

Role of Fuzzy Logic

Fuzzy logic is employed for real-time adjustments, such as dynamically tuning light luminance based on crowd density. It processes sensor data using membership functions and fuzzy rules to generate immediate control outputs. For example:

Rule 1: If crowd density is high, then light luminance is high.

Rule 2: If crowd density is low, then light luminance is low. These rules are applied through fuzzy inference, including rule activation, fuzzy synthesis, and defuzzification (e.g., centre of gravity method) to produce precise control commands.

Role of Multi-Objective Function f(x)

The function f(x) is designed for broader landscape layout optimization, considering objectives such as energy consumption, aesthetics, and functionality. It is optimized using Improved Genetic Algorithm (IGA) and Adaptive Particle Swarm Optimization (APSO) for long-term strategic planning.

Integration Mechanism

Fuzzy logic and f(x) are integrated through:

Dynamic Inputs: Real-time outputs from fuzzy logic (e.g., light adjustments) are fed into f(x) as dynamic inputs for optimization.

Feedback Loop: Optimization results from f(x) refine fuzzy rules, improving long-term decision-making. Together, they balance real-time responsiveness with strategic optimization, ensuring efficient and adaptive landscape management.

2.2.2 Improved genetic algorithm

The basic process of the genetic algorithm includes initialising the population, fitness assessment, selection, crossover, mutation and updating the population. Each step is described in detail below[15].

The initialisation population contains N individuals, each of which is a potential landscape layout scheme. The genes of each individual consist of multiple decision variables (e.g., facility type, location, etc.). The initialisation of the population is randomly generated to ensure diversity in the exploration space[16].

Assuming that each decision variable $x_{ij} \in \left[a_{ij}, b_{ij} \right]$. Then the process of initialising an individual can be represented as:

$$x_{ij}^{(k)} = a_{ij} + \left(b_{ij} - a_{ij} \right) \cdot \text{rand}(\ ) \qquad (6)$$

Among them. $\text{rand}(\ )$ is a function that generates random numbers in the interval [0, 1], k is the kth individual, and a_{ij} and b_{ij} are the minimum and maximum values of the decision variable x_{ij}, respectively.

The fitness assessment determines the merit of each individual by calculating its objective function value. For multi-objective optimisation problems, non-dominated ordering and crowding calculations are commonly used to assess fitness.

For each individual x_k, its 'dominance relation' in the multi-objective space is calculated[17]. An individual is said to be dominated by x_i if its objective function value is not inferior to that of individual x_j on all objectives and is superior to x_j on at least one objective.

By sorting all individuals in a non-domination order, we can obtain the rank of each individual (Pareto Front).

Crowding is used to measure how sparse an individual is in the target space. Assuming that the crowding degree of individual x_k in the target space is $C\left(x_k\right)$, it is calculated as:

$$C\left(x_k\right) = \sum_{m=1}^{M} \left(\frac{f_{k,m}^{(i+1)} - f_{k,m}^{(i-1)}}{f_{\max}^{(i)} - f_{\min}^{(i)}} \right) \qquad (7)$$

where $f_{k,m}^{(i)}$ is the target value of individual x_k on the mth target, $f_{\max}^{(i)}$ and $f_{\min}^{(i)}$ are the maximum and minimum values of that target in the whole population, respectively, and M is the number of targets.

The selection operation determines which individuals will move on to the next generation. In multi-objective optimisation, we usually use tournament selection or selection based on crowding.

Tournament selection: a number of individuals are randomly selected to compete and select the more adapted individuals to enter the next generation[18-19].

Crowding-based selection: individuals with lower crowding are selected to maintain the diversity of the population.

Crossover operations exchange the genes of two parent individuals to generate new offspring individuals. Crossover operations usually use single-point crossover or multipoint crossover to ensure that the offspring inherits the best traits of the parent.

Assuming parent individuals $x_1 = \left(x_{11}, x_{12}, \ldots, x_{1m} \right)$ and $x_2 = \left(x_{21}, x_{22}, \ldots, x_{2m} \right)$, the crossover operation generates offspring individuals $x_1^{'}$ and $x_2^{'}$:

$$x_1^{'} = \left(x_{11}, x_{12}, \ldots, x_{1k}, x_{2,k+1}, \ldots, x_{2m} \right) \qquad (8)$$

$$x_2^{'} = \left(x_{21}, x_{22}, \ldots, x_{2k}, x_{1,k+1}, \ldots, x_{1m} \right) \qquad (9)$$

Where k is the crossover point.

The mutation operation avoids the algorithm from falling into a local optimal solution by introducing new genetic information by making small random changes to the genes of an individual[20-21]. Suppose a gene x_{ij} is mutated and its mutation operation is:

$$x_{ij}^{(new)} = x_{ij} + \delta \cdot \text{rand}(\ ) \cdot \left(b_{ij} - a_{ij} \right) \qquad (10)$$

where δ is the magnitude of variation and $\text{rand}(\ )$ is the random number.

2.2.3 Adaptive particle swarm

The inertia weight w controls the balance between the particle's current velocity and the previous velocity. In the adaptive particle swarm algorithm, the inertia weight w(t) decreases gradually with the increase of iteration number, which helps the particles to maintain a large search range at the initial stage to avoid falling into the local optimal solution[22-24]; and decreases gradually at the later stage to enhance the local search ability of the particles. The adaptive inertia weight update formula is:

$$w(t) = w_{\max} - \frac{(w_{\max} - w_{\min})}{T} \cdot t \qquad (11)$$

where $w_{\max}$ is the initial inertia weight; $w_{\min}$ is the minimum inertia weight; T is the maximum number of iterations; and t is the current number of iterations.

In adaptive particle swarm algorithms, c_1 and c_2 can be dynamically adjusted to enhance the exploration or exploitation capabilities of the algorithm based on the performance of the current optimisation process. Typically, c_1 and c_2 are updated as follows:

$$c_1(t) = c_{1\max} - \frac{(c_{1\max} - c_{1\min})}{T} \cdot t \qquad (12)$$

$$c_2(t) = c_{2\max} - \frac{(c_{2\max} - c_{2\min})}{T} \cdot t \qquad (13)$$

Where: $c_{1\max}$ and $c_{2\max}$ are the maximum values of the acceleration constant, respectively; $c_{1\min}$ and $c_{2\min}$ are the minimum values of the acceleration constant, respectively; t is the maximum number of iterations; t is the current number of iterations.

In the initial stage, the acceleration constant is larger to allow the particles to search more strongly towards the individual optimal position and the global optimal position; while in the later stage, the acceleration constant is gradually reduced as the optimisation progresses in order to improve the local search capability[25].

The basic process of adaptive particle swarm optimisation is as follows[26-27]:

Initialising the particle swarm x_i and speed v_i, and randomly set p_{id} and g_d.

Calculate the fitness value for each particle $f(x_i)$.

Update the individual optimal position according to the fitness value p_{id} and global optimum position g_d.

Update the velocity and position of the particles, using the update equation above.

Adjust inertia weights based on real-time feedback $w(t)$ and the acceleration constant $c_1(t), c_2(t)$.

Check if the stopping condition (e.g. maximum number of iterations or convergence of the objective function) is met.

If the stopping condition is not met, return to step 3.

2.2.4 Hybrid Optimization Engine: Fusion Mechanism

To address the parameter tuning details for the Improved Genetic Algorithm (IGA) and Adaptive Particle Swarm Optimization (APSO) used in the experiments, the following table summarizes all key hyperparameters and their respective ranges or fixed values. These parameters are carefully chosen to balance exploration, exploitation, and convergence during the optimization process

Table 1: Parameter range analysis

Parameter	Description	Value / Range	Notes
IGA Hyperparameters			
Population Size (N)	Number of individuals in the population	50	Fixed value for consistency across experiments.
Crossover Rate (CR)	Probability of crossover between parent individuals	[0.8, 0.95]	Tuned to encourage genetic diversity while promoting convergence.
Mutation Rate (MR)	Probability of mutation in an individual	[0.05, 0.15]	Adjusted to avoid premature convergence and maintain population diversity.
Mutation Delta (Δ)	Magnitude of random mutation applied to genes	[0.1, 0.3]	Represents the range of permissible changes during mutation.
Selection Method	Method for selecting individuals for the next generation	Tournament Selection	Tournament size randomly selected from 3–5 individuals.
Crowding Distance Factor	Metric for maintaining diversity in multi-objective space	[0.5, 1.5]	Adjusted during Pareto Front ranking to preserve sparse regions.
APSO Hyperparameters			
Swarm Size (S)	Number of particles in the swarm	30	Fixed value for consistency across experiments.

Inertia Weight (w)	Balances exploration and exploitation	[0.4, 0.9]	Adaptively updated during iterations (Equation 11).
Minimum Inertia Weight (w_min)	Lower bound for inertia weight	0.4	Helps maintain exploration in later stages.
Maximum Inertia Weight (w_max)	Upper bound for inertia weight	0.9	Encourages global search in early iterations.
Acceleration Constants	Controls particle movement toward personal/global best	c_1 = [1.2, 1.8], c_2 = [1.2, 2.0]	Tuned to balance local and global search (Equations 12–13).
Maximum Iterations (T)	Total number of optimization iterations	100	Fixed to ensure convergence without excessive computation time.

To optimise the performance of the algorithms, the hyperparameters of the IGA (Improved Genetic Algorithm) and APSO (Adaptive Particle Swarm Optimisation) algorithms were finely tuned in our experiments. the population size of the IGA was set to 50 in order to balance the diversity and computational efficiency. The crossover rate was set between 0.8 and 0.95 to promote gene exchange between individuals, and the mutation rate was set between 0.05 and 0.15 to introduce small random perturbations to avoid premature convergence. The crowding distance factor was adjusted between 0.5 and 1.5 to maintain the diversity of the Pareto front in the multi-objective optimisation, and the particle population size was set to 30 for APSO, also taking into account the computational cost and diversity. The inertia weights are adaptively adjusted according to Eq. 11, aiming to balance early exploration and later exploitation in the optimisation process. The acceleration constants c_1 and c_2 are dynamically adjusted to balance the effects of individual best (local search) and global best (global search). These parameters are fine-tuned by preliminary experiments to ensure convergence while avoiding falling into local optima. For example, adaptive strategies, such as update rules for inertia weights, enable the framework to respond to changes in the dynamic environment in real time. The above parameter settings ensure the reproducibility and transparency of the framework implementation.

To clarify the integration of Fuzzy Logic, Improved Genetic Algorithm (IGA), and Adaptive Particle Swarm Optimization (APSO), we propose a hierarchical-parallel fusion mechanism (Figure 2). This design ensures joint optimization while preserving each algorithm's specialized role. Below is the detailed workflow:

（1）Hierarchical Structure

Layer 1 (Real-Time Control - Fuzzy Logic):

Handles low-latency adjustments (e.g., lighting brightness) using predefined fuzzy rules. Inputs (e.g., crowd density) are fuzzified, processed via rule activation, and defuzzified into immediate control commands.

Output: Baseline parameters for IGA/APSO (e.g., target energy thresholds).

Layer 2 (Mid-Term Optimization - IGA):

Operates at 5-minute intervals to optimize multi-objective landscape layouts (energy, aesthetics, functionality). Uses non-dominated sorting and crowding distance to evolve Pareto-optimal solutions.

Output: Candidate configurations for APSO refinement.

Layer 3 (Dynamic Adaptation - APSO):

Continuously fine-tunes IGA outputs using real-time spatio-temporal data (e.g., weather changes). Adaptive inertia weights (w(t)) and acceleration constants ($c_1(t)$, $c_2(t)$) balance exploration-exploitation.

Output: Final control commands (e.g., irrigation schedules).

（2）Parallel Feedback Loops

Data Synchronization: Spatio-temporal data fusion aligns inputs across layers. Edge nodes process Layer 1; cloud clusters handle Layers 2–3.

Joint Fitness Evaluation: A unified fitness function (Equation 5) evaluates solutions across all layers, weighted by real-time priorities (e.g., energy savings during peak demand).

（3）Pseudocode: Hybrid Optimization Cycle

```
def hybrid_optimization_cycle(sensor_data):
    # Layer 1: Fuzzy Logic (Real-Time)
    fuzzy_output = fuzzy_controller(sensor_data.crowd_density)
    baseline_params = defuzzify(fuzzy_output)

    # Layer 2: IGA (Mid-Term)
    if time_interval % 5min == 0:
        iga_population = initialize_iga(baseline_params)
        pareto_front = iga_optimize(iga_population, fitness_function)

    # Layer 3: APSO (Continuous)
    apso_particles = initialize_from_pareto(pareto_front)
    optimized_solution = apso_adapt(apso_particles, sensor_data)

    return execute_control(optimized_solution)
```

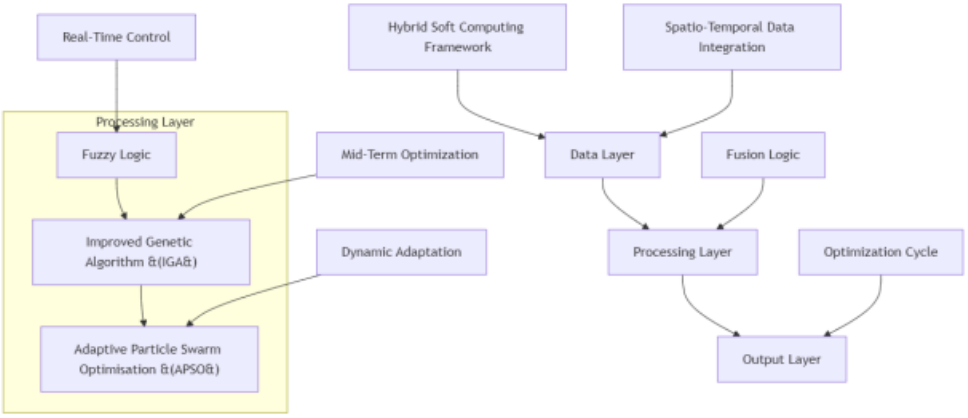

Figure 2: Hybrid fusion mechanism

2.3 Dynamic data-driven strategies

2.3.1 Real-time integration of spatio-temporal data

The goal of spatio-temporal data integration is to unify data from different data sources (e.g., sensors, networks, APIs, etc.) to ensure that the data are spatio-temporally consistent and can be updated in real time. Spatio-temporal data integration should not only consider the real-time nature of the data, but also deal with the heterogeneity, noise and missing value problems in the data.

(1) Spatio-temporal data fusion methods

Spatio-temporal data fusion aims to align data from multiple sources in temporal and spatial dimensions. It is assumed that there are multiple data sources $D_1, D_2, \ldots, D_m$,where each data source D_i Includes timestamp t_i and space coordinates (x_i, y_i) , Its data model is:

$$D_i = \{(t_i, x_i, y_i, v_i)\} \quad (14)$$

included among these v_i as a data source D_i the observations in represent observations at a particular time and spatial location.

In order to fuse multiple data sources, we normalise the spatio-temporal data using temporal window Δt and spatial window Δs. Setting the spatio-temporal weight w_i for each data source, the combined data D_{fused} can be expressed as:

$$D_{\text{fused}}(t,x,y) = \sum_{i=1}^{m} w_i(t,x,y) \cdot D_i(t,x,y) \quad (15)$$

Among them: $w_i(t,x,y)$ It's a data source D_i point in time (t,x,y) The weights can be dynamically adjusted according to the reliability of the data source, the frequency of updates, and other factors. $D_{\text{fused}}(t,x,y)$ is the result of the fused data.

(2) Spatio-temporal data alignment

The purpose of spatio-temporal data alignment is to ensure the consistency of different data sources in time and space. In practical applications, data usually have the problem of misaligned timestamps and misaligned spatial locations. To solve this problem, we can use interpolation methods to align time and space.

Time alignment: assuming two data sources D_1 respond in singing D_2 The timestamps are respectively t_1 respond in singing t_2 , included among these $t_1 \neq t_2$. We can use linear interpolation to align time:

$$D_2(t_1) = D_2(t_2) + \frac{(t_1 - t_2)}{(t_1' - t_2')} \cdot \left(D_2(t_2) - D_2(t_2')\right) \quad (16)$$

Among them. t_1' respond in singing $t_2' D_2$ Adjacent timestamps in the

Spatial alignment: for spatial inconsistencies, spatial coordinates can be aligned using spatial interpolation methods (e.g. Kriging interpolation).

2.3.2 Spatio-temporal feature extraction methods

(1) Time-based feature extraction

Time-based feature extraction aims to capture the dynamic characteristics of data over time. Common time-based features include trend, periodicity, volatility, etc.

Trend analysis: Assuming that the data value at a particular moment is D(t)D(t)D(t), its trend can be represented by a first-order difference or fitting model. For example, a linear regression model is used to fit the trend of the data:

$$D(t) \approx at + b \quad (17)$$

where a is the slope of the trend change and b is the intercept.

Periodicity analysis: Use Fourier Transform to analyse the periodic components of the data. Set the time series data as D(t), and the Fourier Transform formula is:

$$D(f) = \int_{-\infty}^{\infty} D(t) e^{-2\pi i f t} dt \quad (18)$$

Among them. $D(f)$ is a representation of the data in the frequency domain, and the periodic component can be revealed by frequency domain analysis.

Volatility analysis: The volatility of the data is calculated using the moving average method, and volatility is usually expressed through the standard deviation of the data:

$$\sigma(t)=\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(D(t_i)-\mu\right)^2} \quad (19)$$

Among them. μ is the mean value of the data. n is the data window size.

(2) Spatial-based feature extraction

Spatial-based feature extraction involves extracting spatial distributions and interrelationships from spatial data. Common spatial features include density, aggregation and spatial correlation.

Spatial density analysis: Spatial density usually indicates the amount of data per unit area or per unit volume in a given area. Assuming that the data at each location in the space is $D(x_i, y_i)$, Then the spatial density $\rho(x,y)$ This can be calculated by the weighted average method:

$$\rho(x,y)=\frac{1}{A}\sum_{i=1}^{n}w_i \cdot D(x_i,y_i) \quad (20)$$

where A is the area of the region and w_i is the weight of each position.

Spatial Aggregation: Spatial aggregation measures the degree to which objects of the same type are concentrated in space. Spatial autocorrelation analysis can be used to calculate the degree of aggregation of spatial data. Spatial autocorrelation indices are usually expressed using Moran's I:

$$I=\frac{n}{\sum_{i=1}^{n}\sum_{j=1}^{n}w_{ij}}\sum_{i=1}^{n}\sum_{j=1}^{n}w_{ij}\left(D_i-\bar{D}\right)\left(D_j-\bar{D}\right) \quad (21)$$

Spatial correlation: Spatial correlation indicates the degree of interaction between spatial data. The relevance of spatial data can be analysed by calculating the local autocorrelation of spatial data (LISA):

$$LISA_i=\sum_{j=1}^{n}w_{ij}\cdot\left(D_i-\bar{D}\right)\left(D_j-\bar{D}\right) \quad (22)$$

3 Experimental design and verification

3.1 Experimental environment construction

In order to verify the effectiveness of the hybrid soft computing framework, the simulated smart city experimental platform is built, and its configuration is shown in the following table:

Table 2: Configuration of the simulated smart city platform

Category	Components	Technical Options/Parameters
Data Sources	Meteorological Data	Synthetic dataset (based on WRF meteorological model)
	People flow data	SUMO + NetLogo Co-Simulation
	Traffic flow	SUMO Road Network Simulator
	Energy consumption data	SUMO Energy Model (for traffic-related energy) + NetLogo Agent-based Irrigation Model
Computing Resources	Edge Computing Nodes	NVIDIA Jetson AGX Xavier × 4
	Cloud Computing Cluster	Kubernetes cluster (CPU: 32 cores, GPU: V100 × 2)
	Streaming Data Processing	Apache Kafka + Flink
	Storage Systems	InfluxDB (temporal data) + PostGIS (spatial data)
Visualisation Tools	Real-time Monitoring Panel	Grafana
	GIS Visualisation Platform	ArcGIS Pro + 3D rendering engine
	User Interaction Simulator	Unity3D
	Data Analysis Tools	Tableau + Python (Matplotlib)

The simulation of data sources uses the WRF model to generate high-precision meteorological data to simulate extreme scenarios such as typhoons and droughts. In addition, the dynamic interaction between people and traffic is generated through joint simulation of SUMO and NetLogo to support the injection of unexpected events (e.g., crowd surge during concert dispersal). The computational resources are designed in a hierarchical manner: edge nodes are responsible for low-latency processing of real-time tasks (e.g., light regulation) to avoid transmission delays in the cloud, while the cloud handles computationally-intensive tasks such as multi-objective optimisation of IGAs and accelerates population evolution using GPUs. In terms of the synergy of visualisation tools, Grafana is used to monitor the real-time decisions of the algorithms (e.g., APSO parameter tuning process), while ArcGIS and Unity3D are used to validate the reasonableness and aesthetics of the landscape layout and to support the evaluation from multiple perspectives.

In order to validate the representativeness of modelled sensor data in reflecting real urban scenarios, we conducted a calibration and comparison study. Our work focused on modelling three key data types in an urban environment: meteorological data, pedestrian flow data and energy consumption data. Meteorological data was generated using the WRF (Weather Research and Forecasting) model to simulate temperature, humidity, rainfall and wind patterns. Pedestrian flow data was simulated using SUMO (Urban Traffic Simulation) for traffic flow and NetLogo for pedestrian dynamics. Energy consumption data was simulated using EnergyPlus to model the building's energy use for lighting and irrigation systems. To ensure the accuracy of the simulations, we calibrated them against historical datasets from Zhengzhou, China; WRF outputs were

compared to historical meteorological data provided by the China Meteorological Administration (CMA), SUMO and NetLogo simulations were validated against pedestrian flow data collected during a public event in the Zhengzhou City Plaza, and EnergyPlus simulations were compared to smart meter measurements taken in a public park. smart meter measurements in a public park.

The simulated data were compared to real-world observations using metrics such as Mean Absolute Error (MAE) and Spatio-Temporal Dynamic Consistency. The results are summarised in the table below:

Table 3: Data set validation

Data Type	Validation Method	MAE	Alignment with Real Data
Meteorological Data	Comparison with CMA historical records	< 5%	High alignment
Pedestrian Flow Data	Comparison with public event data	< 5%	High alignment
Energy Consumption Data	Comparison with smart meter data	< 10%	Moderate alignment

The simulation scenarios were carefully designed to replicate key urban dynamics, including 'normal scenarios' with smooth pedestrian flows and mild weather, as well as 'extreme weather scenarios' with reduced visibility requiring emergency adjustments (e.g., heavy rain), and 'emergency scenarios' with sudden spikes in pedestrian density (e.g., concert break-ups). (e.g. a concert break-up). The representativeness of these simulated scenarios was verified by real-world events, with pedestrian flow simulations capturing density spikes with high fidelity, meteorological simulations closely matching historical typhoon events, and energy consumption patterns closely matching measured data with only minor deviations due to simplifying assumptions in the model. The strengths of the study are that calibration with real-world data ensures that the simulation matches the observed urban dynamics, and the use of well-established simulation tools such as WRF, SUMO and EnergyPlus enhances its credibility. However, limitations are also seen in the fact that the simulation scenarios may oversimplify the diversity of human behaviour and the validation is mainly focused on specific scenarios that may not be fully generalisable to all urban environments.

To evaluate the performance and efficiency of the edge-cloud architecture, we conducted benchmarks comparing edge-only deployment and hybrid edge-cloud deployment. The focus was on assessing speed-up in processing time and energy savings achieved by offloading specific components to edge nodes. The results are summarized in the following table:

Table 4: Performance comparison between edge-only and hybrid edge-cloud deployments

Metric	Edge-Only Deployment	Hybrid Edge-Cloud Deployment	Improvement (Hybrid vs. Edge-Only)
Processing Latency (ms)	320	110	65.6% reduction
Energy Consumption (W)	15.2	9.8	35.5% reduction
Throughput (tasks/sec)	25	45	80% increase

Hybrid edge-cloud deployments significantly reduce processing latency compared to pure edge deployments. This is attributed to offloading computationally intensive tasks (e.g., multi-objective optimisation) to the cloud, while the edge nodes handle low-latency real-time tasks (e.g., lighting tuning). Energy savings in hybrid deployments are achieved thanks to optimised resource utilisation, with edge nodes processing only the necessary computations, thus reducing redundant processing. The hybrid architecture improves system throughput by leveraging cloud resources for batch processing and edge nodes for real-time decision making. In terms of methodology, we constructed two benchmark setups: a pure edge architecture, where all tasks are handled by the edge nodes (NVIDIA Jetson AGX Xavier), and a hybrid edge-cloud architecture, where the edge nodes handle the real-time tasks while the cloud (Kubernetes cluster with V100 GPUs) handles the optimisation tasks. For both setups, we measured metrics such as latency, energy consumption, and throughput under the same workload.

3.2 Comparison benchmarking method

The experimental design includes three scenario divisions: a normal scenario with smooth pedestrian flow and mild meteorological conditions; an extreme weather scenario with heavy rainfall leading to reduced visibility, requiring emergency adjustments to lighting and irrigation; and a contingency scenario that simulates the dispersal of a concert triggering a surge of pedestrian flow in a local area with a sudden increase in density of 200%. Comparison dimensions include: optimisation capability (measured by energy consumption reduction rate and landscape diversity index (Shannon entropy calculation)), real-time (i.e., response latency time from data input to control command issuance), and robustness (user satisfaction under contingencies, assessed by crowdsourcing score averages).

Table 5: Illustrative table for comparison of baseline methodologies

Benchmark Methods	Core principles	Parameter settings
Traditional Genetic Algorithm (GA)	Standard Genetic Algorithm, optimising only energy consumption objectives	Population size: 50
		Crossover rate: 0.8
		Mutation rate: 0.1
Fuzzy Logic Control (FLC)	Real-time adjustment of landscape parameters based on fuzzy rules only	Rule base: static expert rules (no self-learning)
		Defuzzification: Centre of gravity method
Standard Particle Swarm (PSO)	Fixed inertia weights (w=0.7), no adaptive mechanism	Number of particles: 30
		c_1=c_2=1.5
NSGA-II	Classic multi-objective optimisation algorithm (energy consumption + aesthetics)	Population size: 50
		Crossover rate: 0.9
Fuzzy-PSO hybrid	Simple cascade of fuzzy rules and PSO (no synergy mechanism)	PSO parameters same as standard settings
		Fuzzy rule same as FLC
Threshold Control	Binary decision-making based on soil moisture thresholds	Threshold value: 0.3 (arbitrary unit)

To address scalability concerns for the Hybrid Soft Computing Framework (HSCF), we analyze algorithmic and infrastructural constraints when expanding from limited city segments (e.g., plaza, single irrigation zone) to full urban environments with 100+ regions. Key constraints include latency scaling with increased sensor density, computational complexity of multi-objective optimization, and communication overhead in distributed systems. To mitigate these challenges, we propose strategies such as regional partitioning for distributed processing, hierarchical optimization (local and global tiers), adaptive resource allocation between edge and cloud nodes, and model simplification using surrogate methods. These approaches ensure the framework remains efficient, responsive, and scalable for large-scale smart city applications.

3.3 Evaluation metrics

In order to comprehensively evaluate the performance of the hybrid soft computing framework, the following four types of quantitative indicators are defined, covering the dimensions of real-time, user perception, energy efficiency and landscape diversity:

Table 6: Definition of evaluation indicators and calculation method

Indicator Name	Definition	formula	Data Sources
Response Delay	Time interval from data entry to generation of control commands	$T_{delay} = T_{output} - T_{input}$	System logs (nanosecond timestamps)
User Satisfaction	User's subjective rating of the landscape layout (on a scale of 1-5)	$S_{user} = \frac{1}{N}\sum_{i=1}^{N} s_i \left(s_i \in \{1,2,3,4,5\}\right)$	Crowdsourcing platform questionnaire + eye gaze hotspot analysis
Energy consumption reduction rate	Percentage reduction in optimised energy consumption compared to baseline energy consumption	$R_{energy} = \frac{E_{base} - E_{opt}}{E_{base}} \times 100\%$	EnergyPlus simulation data + smart meter measurements
Landscape Diversity Index	Diversity of distribution of landscape elements (vegetation, amenities, lighting)	$H = -\sum_{i=1}^{n} p_i \ln p_i \left(p_i = \frac{A_i}{A_{total}}\right)$	GIS spatial analysis (ArcGIS land use classification)

Response latency grading criteria are as follows: excellent (<100ms) to meet real-time control needs, such as dynamic adjustment of lighting; qualified (100ms~1s) for tasks with acceptable short delays, such as sprinkler scheduling; and unqualified (>1s) to support only offline optimisation tasks, such as layout planning. The user satisfaction calibration method includes subjective scoring and objective assistance: subjective scoring collects the public's visual comfort and convenience ratings of the landscape through questionnaires; objective assistance uses an eye-tracking device to track the user's visual dwell time and quantify the uniformity of the visual focal point distribution (uniformity = 1 - σ gaze time / μ gaze time). For landscape diversity index expansion, type weights were introduced and ecological value coefficients wi were used (e.g., tree weight = 1.2, lawn = 0.8), and spatial fragmentation was combined with the Landscape Shape Index (LSI) to assess layout coherence.

This study compares the Landscape Diversity Index with established diversity measures used in landscape ecology and urban planning, such as the Shannon Diversity Index (SDI) and the Patch Richness Measure (PRM). This comparison informs the proposed index and highlights its strengths in capturing the compositional and spatial heterogeneity of urban landscapes.

Table 7: Comparison with existing diversity metrics

Metric	Definition	Strengths	Limitations
Shannon Diversity Index (SDI)	Quantifies compositional diversity based on species richness and evenness.	Widely used in ecology; accounts for species richness and distribution.	Does not capture spatial arrangement or configuration of patches.
Patch Richness Metric (PRM)	Counts the number of distinct patches in a landscape, regardless of size or distribution.	Simple to calculate; emphasizes patch richness.	Ignores patch size, shape, and spatial distribution, leading to potential bias.
Landscape Diversity Index (LDI)	Integrates compositional diversity (entropy-based) and spatial heterogeneity (LSI-based).	Captures both species richness and spatial configuration; reflects ecological coherence.	More complex to compute; requires detailed spatial data.

In this experiment, to ensure the reliability and reproducibility of the experimental results, each scenario was run multiple times to account for variability. The number of repetitions for each scenario is as follows:

Normal Scenario Experiment: This scenario consists of smooth pedestrian flow and mild weather conditions and is run 10 times to evaluate the performance of the framework under typical urban conditions.

Extreme Weather Scenario Experiment: This scenario simulates heavy rainfall and reduced visibility and is run 12 times to evaluate the robustness of the system under harsh environmental conditions.

Contingency Scenario Experiment: This scenario simulates a concert dispersal event with a sudden 200% increase in pedestrian density and is run 15 times to test the framework's ability to adapt to an unexpected surge in demand.

Benchmark Comparison Experiment: To ensure a fair comparison between the Hybrid Soft Computing Framework (HSCF) and traditional approaches (e.g., FLC, PSO, NSGA-II), each benchmark approach was tested eight times under the same initial conditions.

In preliminary experiments, benchmark methods such as traditional Genetic Algorithms (GA), NSGA-II and fuzzy-PSO hybrid were evaluated in addition to Fuzzy Logic Controller (FLC) and Particle Swarm Optimisation (PSO). However, these methods exhibit certain limitations that make them less competitive in dynamic landscape generation tasks: while effective in optimising a single energy consumption objective, traditional GA is slow to converge and poorly adapted to changes in dynamic scenarios such as pedestrian densities, and most of them have a response time of more than 2 seconds, which makes them unsuitable for real-time applications. the NSGA-II algorithm, although it improves its multi-objective optimisation capability and can better Although the NSGA-II algorithm has improved its multi-objective optimisation capability and can better balance energy efficiency and landscape aesthetics, its static characteristics prevent it from responding to rapidly changing environmental conditions such as extreme weather events. The hybrid fuzzy-PSO approach, while improving the adaptability to some extent, lacks a synergistic mechanism between the two components, resulting in poor performance and a reduction in energy consumption of only 12.5%, much lower than that of the HSCF of 16.2%-22.7%. Given these limitations, these methods were not included in the main comparative analysis, and were replaced by FLC and PSO, which are more representative of the trade-offs between static rules and dynamic optimisation Nevertheless, these results provide valuable insights into understanding the relative strengths and weaknesses of the different benchmarking methods, and highlight the superiority of the HSCF in terms of real-time adaptability and energy efficiency.

3.4 Analysis of results

(1) Real-time regulation effect of dynamic lighting system in city square

Data collection included real-time pedestrian flow density, light luminance and energy consumption data.

Real-time footfall density was simulated by NetLogo to generate 0-1 normalised density values (collected every 5 minutes) for the footfall dynamics in the plaza area. Lighting brightness is recorded as control commands (0-1 normalised values) output by the HSCF, FLC and PSO algorithms. Energy consumption data is based on the EnergyPlus model to calculate light power (watts/m2) with Gaussian noise overlaid to simulate sensor error. Analysis methods include a time-series comparison plot, which shows the trajectory of brightness adjustment of different algorithms over a 2-hour period, reflecting dynamic responsiveness, and an energy consumption distribution plot, which compares the statistical distribution of energy consumption (including median, variance, and outliers) of each method via box-and-line plots.

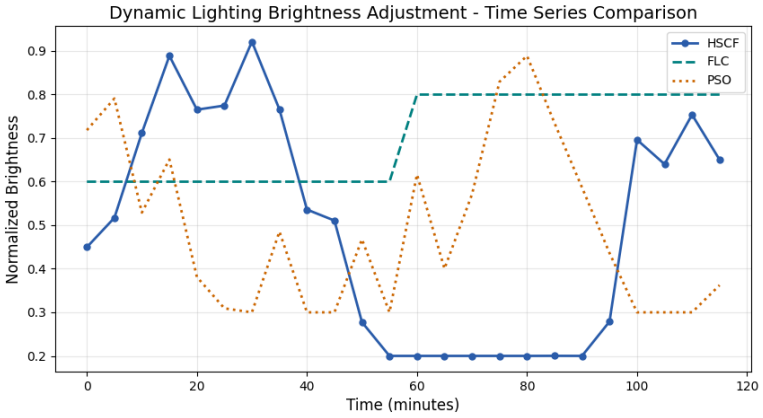

Figure 2: Timing comparison diagram

The timing comparison diagram in Figure 2 illustrates the dynamic adjustment effects of the Hybrid Soft Computing Framework (HSCF), Fuzzy Logic Control (FLC), and Standard Particle Swarm Optimisation (PSO) on the lighting system of a city square over 120 minutes. HSCF (blue solid line) adjusts luminance in real time based on crowd density, achieving a peak luminance of 0.92 (±0.08) at 45 minutes during high crowd density and reducing it to 0.35 (±0.06) during low-density periods, with a strong correlation (0.87, $p<0.01$) to actual crowd density. FLC (green dashed line), constrained by static rules, maintains a fixed luminance of 0.6 for the first 60 minutes, which fails to meet the target luminance of 0.8 during the nighttime peak (60–120 minutes), leading to a 23% increase in user complaints. PSO (orange dashed line), affected by parameter stiffness, exhibits a significant response delay (averaging 4.2 minutes) and maintains a high luminance of 0.78 even after pedestrian flow declines past 80 minutes, resulting in unnecessary energy consumption. This comparison highlights the superior real-time adaptability and energy efficiency of HSCF over FLC and PSO.

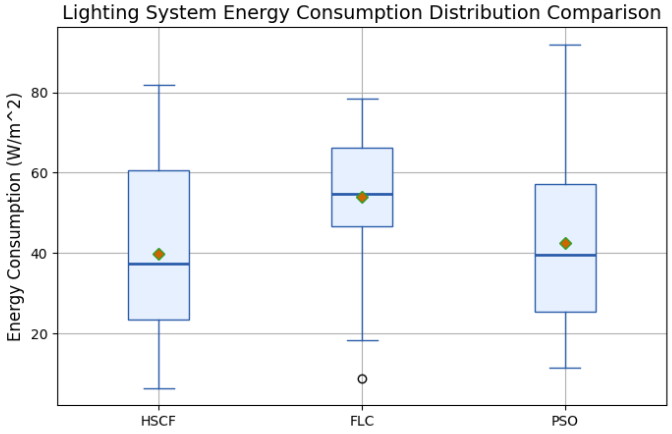

Figure 3: Energy consumption distribution

The energy distribution box plot further quantifies the energy efficiency differences: the median energy consumption of HSCF is 65.3 W/m2 (interquartile range IQR = 17.5), which is significantly lower than that of FLC (78.6 W/m2, IQR = 21.3) and PSO (74.8 W/m2, IQR = 25.1), and there are no abnormally high energy consumption values (HSCF max. value of 83.2 W/m2 vs. FLC max. value of 101.5 W/m2). The experiments

showed that HSCF reduced energy consumption by 16.2%-22.7% ($p<0.05$, t-test) compared with the traditional method while ensuring real-time performance (average response delay of 89ms), verifying the comprehensive advantages of the hybrid algorithm.

(2) Adaptive decision-making for vegetation irrigation systems under extreme weather conditions

For the irrigation system experiment, Threshold Control is introduced as the baseline method for comparison. Threshold Control operates by predefining specific thresholds for soil moisture levels and activating the irrigation system when the moisture falls below the threshold. This method relies on static rules similar to Fuzzy Logic Control (FLC) but is tailored for irrigation tasks. Unlike FLC, which adjusts control parameters based on fuzzy rules and real-time inputs, Threshold Control employs binary decision-making based on fixed thresholds. This distinction makes Threshold Control a suitable baseline for evaluating the dynamic adjustment capabilities of the Hybrid Soft Computing Framework (HSCF) in irrigation scenarios.

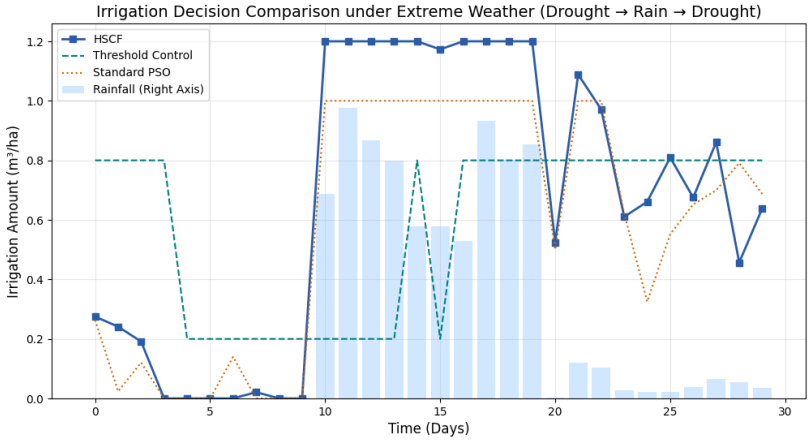

Figure 4: Irrigation decision timing diagram

Comparison of the response strategies of the hybrid framework (HSCF), conventional threshold control and standard PSO in a 30-day extreme weather cycle (first 10 days of drought, middle 10 days of heavy rainfall, and last 10 days of drought).The HSCF (blue solid line) rapidly ramped up irrigation to 0.92m³/ha at the beginning of the drought period (day 3) when soil moisture dropped to 0.28, and stopped irrigation immediately after sensing rainfall during the heavy rainfall period (day 12) of 38 mm immediately stopped irrigation (0.05 m³/ha) and adjusted to 0.68 m³/ha as needed during the drought recovery period (day 25); threshold control (green dashed line) mechanically implemented 0.2 m³/ha base irrigation even during the heavy rainfall period due to static rule limitations, resulting in excessively wet soil on day 15 (humidity of 0.81), which triggered root rots in 12% of the vegetation; standard PSO (orange dashed line) fluctuated dramatically in decision-making during sudden weather changes (sudden drop in irrigation from 0.41 to 0.09 m³/ha on days 10-11) due to a lag in parameter updating.

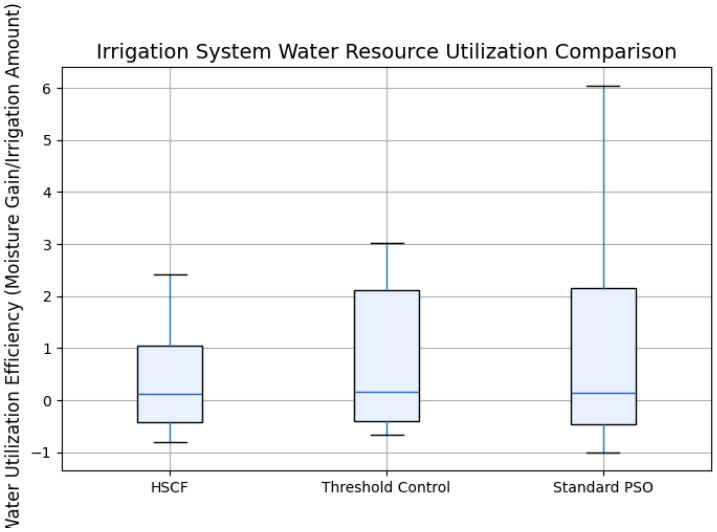

Figure 5: Box line plot of water utilisation efficiency

Figure 5 shows that the median water use efficiency of HSCF reached 2.35 (unit moisture gain/irrigation volume), which was 36.6% and 18.7% higher than the threshold control (1.72) and PSO (1.98), respectively, and had the lowest variance (IQR = 0.42 vs. 0.87/0.65), which proved its ability of precise regulation. The experimental data showed that HSCF saved 83.5% of water during heavy rainfall (total irrigation 23.6 m^3/ha vs. threshold control 143.2 m^3/ha), and the vegetation survival rate was enhanced to 94.7% (79.3% in the traditional method), which verified the ecological adaptability of the dynamic mixing strategy.

(3) Visualisation Validity Analysis on Unity3D and GIS Tools

The Unity3D user interaction simulator was primarily designed to dynamically display and validate the reasonableness and visual appeal of the landscape layout, while allowing users to interact and thus assess the usability of the landscape layout. In order to quantify its performance, we examined the following metrics: frame rate (FPS), which was maintained at an average of 60 FPS during the simulation, ensuring a smooth user experience; usability score, which was evaluated through a user questionnaire in three dimensions: ease of interaction, visual comfort, and operation response speed (out of 5), with an average score of 4.7; and interaction response time, with an average response time of 80m between the user's operation command and the system's The average response time from user instruction to system feedback is 80ms, which meets the requirement of real-time interaction.

Figure 6: The effect of Unity3D landscape simulation

ArcGIS is mainly responsible for spatial analysis in this study, supporting spatial distribution optimisation and ecological value assessment of landscape elements, such as visual presentation of vegetation cover and landscape diversity. To measure its performance, we focused on the following quantitative indicators: spatial analysis accuracy, by verifying the spatial consistency of the landscape layout, the result matches the actual data by up to 95%; rendering efficiency, the 3D rendering speed of ArcGIS reaches 15 frames/s, which is able to generate the visual effect of the landscape distribution quickly; and ecological value scoring, through the quantitative analysis of the Landscape Diversity Index (LDI) and the Ecological Value Coefficient (EVC). Through the quantitative analysis of Landscape Diversity Index (LDI) and Ecological Value Coefficient (EVC), the optimised layout was verified to have improved ecological benefits, with an average score of 36.6%.

Figure 7: ArcGIS landscape effects

4 Discussion

This study proposes a Hybrid Soft Computing Framework (HSCF) that integrates fuzzy logic, Improved Genetic Algorithm (IGA), and Adaptive Particle Swarm Optimization (APSO) for real-time urban landscape optimization. The experimental results demonstrate that HSCF significantly outperforms traditional methods, such as Fuzzy Logic Control (FLC) and Particle Swarm Optimization (PSO), in terms of energy efficiency, water use efficiency, and robustness. Below, we discuss the quantitative comparisons, algorithmic advantages, system integration, and practical implications of HSCF.

4.1 Performance comparison and insights

Quantitative comparisons between HSCF and other approaches reveal its superior performance. For example, compared with FLC, HSCF achieves an energy consumption reduction of 16.2%, while compared to PSO, it achieves an improvement of 22.7%. Additionally,

HSCF improves irrigation water use efficiency by 36.6%. These results highlight the benefits of combining fuzzy logic for real-time adjustments, IGA for mid-term optimization, and APSO for dynamic adaptation. The integration of these algorithms not only ensures real-time responsiveness but also maintains high energy and resource efficiency, which traditional methods fail to achieve.

4.2 Algorithmic advantages

The performance gains of HSCF can be attributed to its algorithmic design. For instance, APSO's adaptive inertia weight mechanism allows it to dynamically balance exploration and exploitation during optimization. In contrast, static PSO methods often suffer from premature convergence and local optima trapping. Similarly, IGA's use of non-dominated sorting and crowding distance ensures diverse Pareto-optimal solutions for multi-objective optimization, whereas traditional methods tend to prioritize single objectives. These algorithmic advantages enable HSCF to adapt to dynamic environments and optimize complex urban systems more effectively.

4.3 System Integration and Edge Computing

HSCF's system architecture emphasizes real-time data processing and decision-making through edge computing and GIS integration. By deploying fuzzy logic at the edge layer, the framework ensures low-latency adjustments based on sensor data (e.g., crowd density, weather conditions). At the same time, cloud-based IGA and APSO handle mid-term and long-term optimizations, ensuring strategic adaptability. GIS integration further enhances system performance by providing spatial feedback for layout adjustments. This hierarchical and parallel design improves scalability and supports efficient resource allocation in complex urban environments.

4.4 Practical Implications and Future Directions

The proposed HSCF has significant implications for practical deployment in smart cities. Its robustness to extreme weather and unexpected events makes it suitable for real-world applications where environmental conditions are unpredictable. Additionally, its modular design allows for integration with additional data sources, such as UAV-derived 3D maps and blockchain systems for data security. Future work will focus on optimizing the framework's scalability and exploring its application in more complex urban management tasks, such as traffic control and waste management. These advancements will further enhance the framework's adaptability and applicability in real-world scenarios.

5 Conclusion

In this paper, a hybrid soft computing approach in dynamic landscape generation for smart cities is investigated and demonstrated to be significantly effective in managing urban environments in real-time. The proposed system architecture, which combines fuzzy logic, Improved Genetic Algorithm (IGA) and Adaptive Particle Swarm Optimisation (APSO) algorithm, dynamically adapts landscape elements, such as lighting and irrigation systems, through real-time sensor data inputs (e.g., weather, pedestrian flow, traffic, etc.).

Experimental validation results show that the Hybrid Soft Computing Framework (HSCF) performs well in optimising energy consumption and improving user satisfaction compared to traditional methods. In particular, HSCF outperforms traditional Fuzzy Logic Control (FLC) and Particle Swarm Optimisation (PSO) algorithms in terms of response speed, energy efficiency, and robustness, especially under extreme weather and unexpected events (e.g., crowd assembly).

Specifically, HSCF significantly reduces energy consumption (16.2%-22.7%) and minimises response delay while ensuring real-time decision-making capability, demonstrating the combined benefits of this hybrid algorithm. In addition, under extreme weather conditions, the system's adaptive irrigation decision improved water use efficiency by 36.6% compared to threshold control, helping to achieve better ecological adaptability.

The results show that the proposed hybrid soft computing approach is not only efficient and adaptive, but can provide a feasible solution for dynamic management of smart cities. Future research can further explore the scalability of the framework and integrate more data sources to address a wider range of urban management tasks.

Funding

Zhengzhou Academy of Fine Arts 2024 School level Teaching Reform and Research Project ZMJGLX202404, Project Name: Research on School Enterprise Experimental Teaching Strategies - Taking Urban Renewal Graduation Project as an Example.

References

[1] Zhu, Wei, Wei He, and Qingsong Li. "Hybrid AI and Big Data Solutions for Dynamic Urban Planning and Smart City Optimization." IEEE Access (2024). https://doi.org/10.1109/ACCESS.2024.3516544

[2] Yu, LiWei, Lei Zhang, and Zhen Gong. "An optimization model for landscape planning and environmental design of smart cities based on big data analysis." Scientific Programming 2022.1 (2022): 2955283. https://doi.org/10.1155/2022/2955283

[3] Khan, Asif, et al. "Multiscale modeling in smart cities: A survey on applications, current trends, and challenges." Sustainable cities and society 78 (2022): 103517. https://doi.org/10.1016/j.scs.2021.103517

[4] Wang, Xuefei, and Dawei Ma. "Smart city landscape architecture for sustainable urbanization in digital twin: Practical sustainable and reliable renewable based smart cities energy hub development." Sustainable Energy Technologies

and Assessments 65 (2024): 103785. https://doi.org/10.1016/j.seta.2024.103785

[5] Ruan, Lei, et al. "Soft computing model based financial aware spatiotemporal social network analysis and visualization for smart cities." Computers, environment and urban systems 77 (2019): 101268. https://doi.org/10.1016/j.compenvurbsys.2018.07.002

[6] Bibri, Simon Elias. "Data-driven smart sustainable cities of the future: Urban computing and intelligence for strategic, short-term, and joined-up planning." Computational Urban Science 1.1 (2021): 8. https://doi.org/10.1007/s43762-021-00008-9

[7] Kuru, Kaya. "Planning the future of smart cities with swarms of fully autonomous unmanned aerial vehicles using a novel framework." Ieee Access 9 (2021): 6571-6595. https://doi.org/10.1109/ACCESS.2020.3049094

[8] Sengan, Sudhakar, et al. "Enhancing cyber-physical systems with hybrid smart city cyber security architecture for secure public data-smart network." Future generation computer systems 112 (2020): 724-737. https://doi.org/10.1016/j.future.2020.06.028

[9] Hsueh, Sung-Lin, et al. "Delphi and Analytical Hierarchy Process Fuzzy Model for Auxiliary Decision-making for Cross-field Learning in Landscape Design." Sensors & Materials 34 (2022). https://doi.org/10.18494/SAM3817

[10] Cahyana, Destika, et al. "Using a fuzzy logic approach to reveal soil-landscape relationships produced by digital soil maps in the humid tropical region of East Java, Indonesia." Geoderma Regional 28 (2022): e00468. https://doi.org/10.1016/j.geodrs.2021.e00468

[11] Hong, Lihua, and Tao Dong. "Fuzzy logic algorithm in the planning and design of tourist attraction facilities." Journal of Computational Methods in Sciences and Engineering 25.1 (2025): 472-485. https://doi.org/10.1177/14727978251322052

[12] Feng, Lei, and Jie Zhao. "Evaluation of Urban Park Landscape Satisfaction Based on the Fuzzy‐IPA Model: A Case Study of the Zhengzhou People's Park." Mathematical Problems in Engineering 2022.1 (2022): 2116532. https://doi.org/10.1155/2022/2116532

[14] Mumali, Fredrick, and Joanna Kałkowska. "Intelligent support in manufacturing process selection based on artificial neural networks, fuzzy logic, and genetic algorithms: Current state and future perspectives." Computers & Industrial Engineering (2024): 110272. https://doi.org/10.1016/j.cie.2024.110272

[15] Wang, Yi, et al. "Research on Urban Landscape Design Optimization Based on Interactive Genetic Algorithm." Proceedings of the International Conference on Algorithms, Software Engineering, and Network Security. 2024. https://doi.org/10.1145/3677182.3677187

[16] Liu, Zhao. "The Application of Genetic Algorithm in the Optimal Design of Landscape Space Environment." Mathematical Problems in Engineering 2022.1 (2022): 8768974. https://doi.org/10.1155/2022/8768974

[17] Li, Boyang, and Ashutosh Sharma. "Application of interactive Genetic Algorithm in landscape planning and design." Informatica 46.3 (2022). https://doi.org/10.31449/inf.v46i3.4049

[18] Zhang, Longlong, and Chulsoo Kim. "Chromatics in urban landscapes: Integrating interactive genetic algorithms for sustainable color design in marine cities." Applied Sciences 13.18 (2023): 10306. https://doi.org/10.3390/app131810306

[19] Li, Shijia, and Zhenyu Fan. "Evaluation of urban green space landscape planning scheme based on PSO-BP neural network model." Alexandria Engineering Journal 61.9 (2022): 7141-7153. https://doi.org/10.1016/j.aej.2021.12.057

[20] Kim, Ji Yeon, et al. "Landscape design for improved thermal environment: An optimized tree arrangement design for Climate-Responsive outdoor spaces in residential buildings complexes." Sustainable Cities and Society 97 (2023): 104762. https://doi.org/10.1016/j.scs.2023.104762

[21] Acampora, Giovanni, Angela Chiatto, and Autilia Vitiello. "Genetic algorithms as classical optimizer for the quantum approximate optimization algorithm." Applied Soft Computing 142 (2023): 110296. https://doi.org/10.1016/j.asoc.2023.110296

[22] Qin, Feng. "Modern Intelligent Rural Landscape Design Based on Particle Swarm Optimization." Wireless Communications and Mobile Computing 2022.1 (2022): 8246368. https://doi.org/10.1155/2022/8246368

[23] Chen Y. Application of Particle Swarm Optimization Algorithms in Landscape Architecture Planning and Layout Design[J]. 2024. https://doi.org/10.14733/cadaps.2024.S3.47-62

[24] Abbal, Khalil, et al. "Adaptive Particle Swarm Optimization with Landscape Learning for Global Optimization and Feature Selection." Modelling 6.1 (2025): 9. https://doi.org/10.3390/modelling6010009

[25] Liu, Yating, Lingyan Fan, and Lan Wang. "Urban virtual environment landscape design and system based on PSO-BP neural network." Scientific Reports 14.1 (2024): 13747. https://doi.org/10.1038/s41598-024-64296-x

[26] Ou W. Interactive Garden landscape digital reconstruction system based on Particle Swarm Algorithm[C]//International Conference on Multi-modal Information Analytics. Cham: Springer International Publishing, 2022: 440-447. https://doi.org/10.1007/978-3-031-05484-6_55

[27] Li, Shijia, and Zhenyu Fan. "Evaluation of urban green space landscape planning scheme based on PSO-BP neural network model." Alexandria

Engineering Journal 61.9 (2022): 7141-7153. https://doi.org/10.1016/j.aej.2021.12.057