Design Studies and Intelligence Engineering
L.C. Jain et al. (Eds.)

doi:10.3233/FAIA231442

Residential Spatial Layout Design Using Advanced Optimization Techniques

Yalong MAO[a], Zhe LAI[b,1], Zitong CHEN[c], Shaojie WANG[d], and Lukui HUANG[e]

[a] *Architectural Design & Research Institute, South China University of Technology, China*
[b] *Senior Engineer, Ph.D., South China University of Technology, Guangzhou, China*
[c] *B.E, Zhongkai University of Agriculture and Engineering, Guangzhou, China*
[d] *Department of Architecture, Ph.D., South China University of Technology, Guangzhou, China*
[e] *Guangxi University of Finance and Economics, Nanning, China*

Abstract: Spatial layout design profoundly influences architectural development, yet traditional manual approaches suffer from limited solutions due to human cognitive constraints and cost-effectiveness consideration. This study explores selected advanced optimization algorithms—Simulated Annealing Algorithm, Dijkstra Algorithm, Frei Otto's Woolthread Model, Slime Mould Algorithm, and Genetic Algorithms—to enhance residential spatial layouts design. Prototypical cases of common residential planning and design challenges were utilized for the simulated experiments. Comparative analysis highlights Dijkstra's Algorithm and the Genetic Algorithm as optimal choices for path optimization and visual adjustments, contributing to the research on residential spatial layouts.

Key words: minimal path, design layout, optimization algorithms, residential planning

1. Introduction

Design layout serves as a foundational element for architectural development. Traditional manual layouts are labor-intensive and tend to yield limited solutions due to cognitive constraints [1]. Automated computational methods, such as optimization algorithms, offer a more efficient and diverse exploration of layouts, mitigating time and cost constraints. However, challenges persist in developing effective computational solutions due to complex site conditions and engineering constraints [2].

The overall floor plan, as an important spatial carrier, often requires cumbersome and repetitive adjustments during the design process. Modern architects turn to residential layout automation for more cost-effective solution to this issue. In this paper, we introduce and compare a series of advanced optimization algorithms which are particularly applicable for optimizing spatial distribution problems in residential planning and layout design. The selected optimization algorithms for automated residential layouts include Simulated Annealing Algorithm, Dijkstra Algorithm, Frei Otto' Woolthread Model, Slime Mould Algorithm, and Genetic Algorithm.

[1] Corresponding author: Zhe Lai; Email: 269442001@qq.com

1.1 Introduction to the selected optimization algorithms

1.1.1 Simulated Annealing Algorithm

Simulated annealing, introduced by Kirkpatrick et al. in 1983[3] as a modification of the Metropolis-Hastings algorithm, is a probabilistic approach for globally optimizing a cost function with multiple local optima. The simulated annealing algorithm has been applied in a wide range of optimization problems in areas such as engineering [4-6] and renewable energy[7]. In urban planning and architecture, Hong (2011) developed a model using simulated annealing for accurate traffic flow forecasting [8]. Sonmez (2008) optimized structural designs for lightweight and high-performance [9]. AlHalawani et al. (2013) generated procedural facade layouts with simulated annealing [10]. Yeh (2006) solved facility layout problems in hospital design by generating optimized floorplans [11] and Michalek et al. (2002) arranged house rooms using a similar technique [12].

1.1.2 Dijkstra's Shortest Path Algorithm

Dijkstra's algorithm, proposed by Edsger Dijkstra in 1959, efficiently finds the shortest path from a source node to all others in a weighted graph. Dijkstra's algorithm has been widely used in areas like network routing [13], transportation[14-15], robotics[16], and video games [17]. In urban planning, Zhong et al. (2022) combined Dijkstra's algorithm with geographic information systems for eco-commuting route recommendations [18]. Within architecture, Sharmeen et al. (2022) optimized evacuation route planning in buildings using Dijkstra's algorithm [19].

1.1.3 Frei Otto' Woolthread Model

Developed by German architect and engineer Frei Otto in the 1970s-1990s, the woolthread model, created at the Institute for Lightweight Structures (ILEK) in Stuttgart, serves as an analog model for optimizing path networks [20-21]. Pioneering the woolthread model's application for generating urban layouts, street networks, and settlement patterns, Otto demonstrated that organic complexity could exhibit inherent optimality and logic. The model challenges the conventional notion of orderly, high-performing, and optimized urban patterns in contrast to modernist grids [22].

1.1.4 Slime Mould Algorithm

The Slime Mould Algorithm, introduced in 2020 [23], has gained considerable attention for its simplicity, strong optimization capabilities, and reliable convergence in addressing diverse real-world problems. Its applications span areas such as traffic prediction [24], path optimization in mobile robots [25], and resource allocation [26].

1.1.5 Genetic Algorithm

Genetic Algorithm (GA) is a special kind of stochastic search algorithm that depicts the biological evolution as the problem-solving technique. GA can be applied for nonlinear programming like traveling salesman problem, minimum spanning tree, scheduling problem. Multiple studies also applied GA to solve complex problems urban design [27-30].

1.2 Project Objective

In the early stages of residential development, layout design involves individual units, regional planning, and road networks. This paper addresses common challenges in residential planning by comparing the performance of the wool-thread model, slime mould algorithm, and Dijkstra's algorithm for on-site building layout. Simulated annealing and genetic algorithms are assessed for view deviation. Using advanced optimization algorithms, we aim to identify optimal strategies for generating residential layouts, translating design concepts into mathematical parameters to drive a generative layout system (Figure 1).

1.3 Optimization methods

This paper adopts the essentials of the optimization algorithms as the solution approach:

a) Clear definition of an objective function for measurement
b) Traverse the variable space to find the optimal value of the objective function
c) Use mathematical or heuristic techniques to accelerate the optimization process
d) Termination conditions to judge whether a satisfactory solution is found

Hence, the prototype problems in residential planning and design to be solved include:

a) Use Voronoi division to determine the basic functions of building objects
b) Traverse these functions using Python
c) Optimize using genetic algorithms, Dijkstra’s algorithm, slime mould algorithm, and wool-thread model.
d) Compare the above algorithms and attempt to propose corresponding optimal strategies and methods (Figure 1).

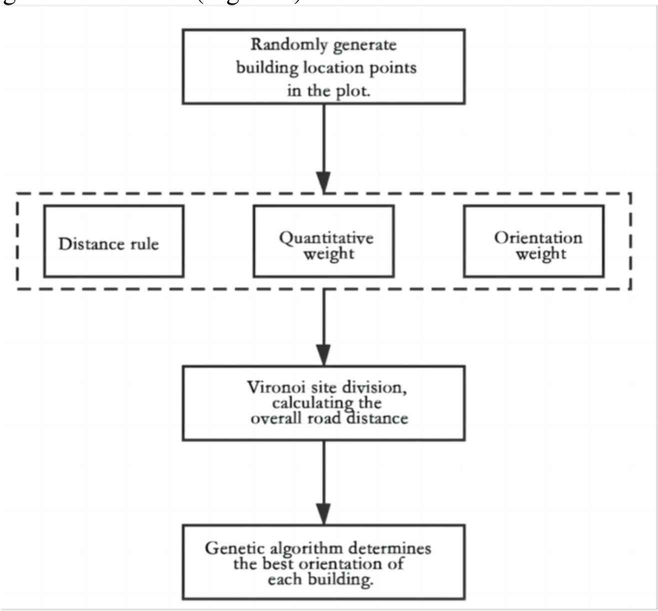
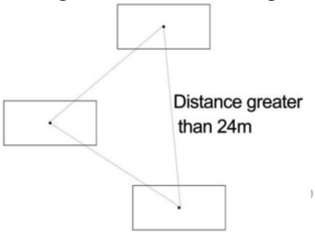


Fig. 1 Illustration of Program Workflow

2. Experiment in Settlement Generation Design

2.1 Distribution of housing units

In urban residential design, individual unit placement is influenced by factors like sunlight, landscape orientation, terrain, and building clusters. Grasshopper generates random building points constrained by Python for proximity checks. Models are evaluated through weighted calculations considering building count, view quality, and distance from exits. Genetic algorithms determine the optimal combination of random seeds and rotation angles for the highest scores, followed by computing the optimal view angle for each building.




```
for pt in pts:
    distances = rs.Distance(pt,pts)
    distances.sort()
    if distances[1] < 24:
        pts.remove(pt)
    else:
        pass
```

Figure 2 Schematic diagram of distance rules

2.1.1 Distance rule. The distribution of residence monomers is mainly based on the distance between the monomers. First of all, it is necessary to maintain the minimum distance between residential buildings. In order to simplify the experiment, the minimum distance of the experiment is set to 24 meters. Secondly, random coordinates points are generated in the red building line by the popular geometry component. Thirdly, in Python, the minimum distance from the central point of every building to the other buildings is traversed. The building shall be removed when the nearest building distance is less than 24 meters. According to this rule, all buildings are traversed, and the resulting building list deviates from the original building set number.

2.1.2 Visual field distance rule. Among the influencing factors of residential orientation, the building view will be the dominant factor in the distribution of buildings under the condition of assuring the north-south and daylight distance. The house's orientation is not usually determined by the optimal direction of sunlight but a compromise result obtained by focusing on sunlight, integrating other landscapes and vision requirements. (Figure 4)

In this generation experiment, the surrounding buildings block the view of the site, combined with the maximum distance of view of the building, in the southeast 45 degrees and 45 degrees southwest to get the best angle of building deflection. (Figure 3)

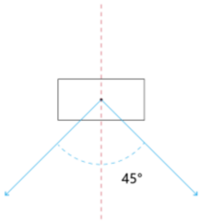


Fig. 3 Sketch map of visual range rules

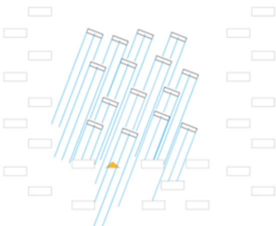


Fig. 4 Sketch map of orientation rules

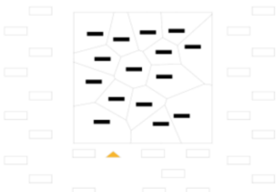


Figure 5 Multi-agent calculation result

2.2. Site division

The integrated factors for residential building design include land use uniformity and convenience. In residential building design, common methods of residential land planning include multi-agent plots design, and environmental zoning. Among them, the Environment of the subdivision is a classic prototype of polygon subdivision in architectural design. It is a continuous polygonal structure that is made up of two parallel bisecting curves. The space between any polygon and other equals that between any point in the polygon and other polygons. The geometry adheres to non-orthogonal building requirements, ensuring uniformity in each building's location. The polygonal grid creates small plot areas with geometric rationality and random beauty. (Figure 5).

2.3. Path optimization and generation

Road planning in residential areas considers hierarchy and convenience. Hierarchy involves primary roads at the residential level, secondary roads at the neighborhood level, and access roads to houses. Road forms are evaluated based on pedestrian flow rates during generative design. Connectivity is crucial, requiring roads to link residence units, neighborhood exits, and key nodes through the shortest paths. Site layout performance is compared using Dijkstra's algorithm (Figure 6), slime mold algorithm (Figure 7), and wool-thread model (Figure 8).

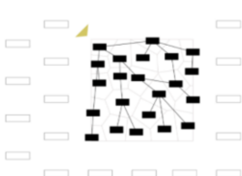


Figure 6 Minimal path using Dijkstra's algorithm

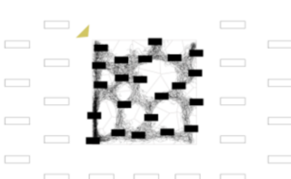


Figure 7 Minimal path using slime mould algorithm

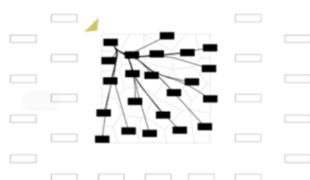


Figure 8 Minimal path using wool-thread model

Comparing three algorithms, Dijkstra's method proves optimal for the minimal path strategy. It addresses the shortest path problem in a weighted graph, calculating a unique shortest path between starting and ending points in a road grid. Evaluating road network throughput and hierarchy involves overlaying the number of shortest paths across segments. Regional planning and path generation are intertwined in residential areas, where experiments assess flow line smoothness along land boundaries. Overlaying road traffic reveals route hierarchy. Figures 7 and 8 show that the slime mold and wool-thread algorithms have higher hierarchical levels and point-to-point overlays than Dijkstra's. Consequently, Dijkstra's is the optimal strategy for shortest path challenges in residential planning and land use integration.

2.4. View Deviation and Generation

To investigate the view deviation, we calculated the total distance of the shortest path from each building to the site entranceand combined with certain weighted building views and number of buildings to evaluate the random building points generated by the Populate Geometry component. By comparing the optimal strategies of the genetic algorithm and simulated annealing algorithm, the optimal strategy for generating random building points is determined.

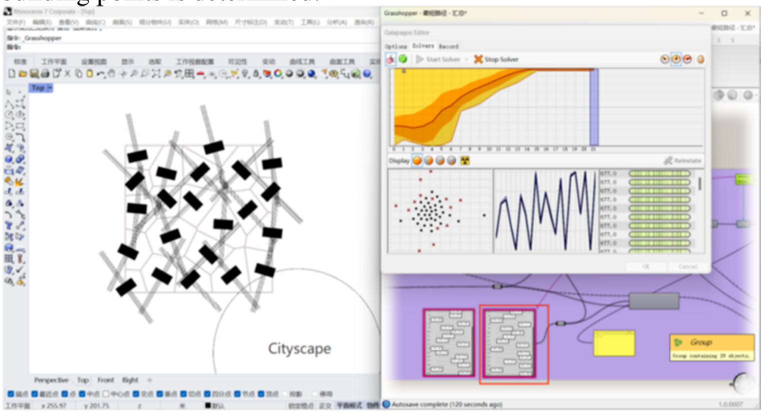


Figure 9 Genetic Algorithm for Random Building Points

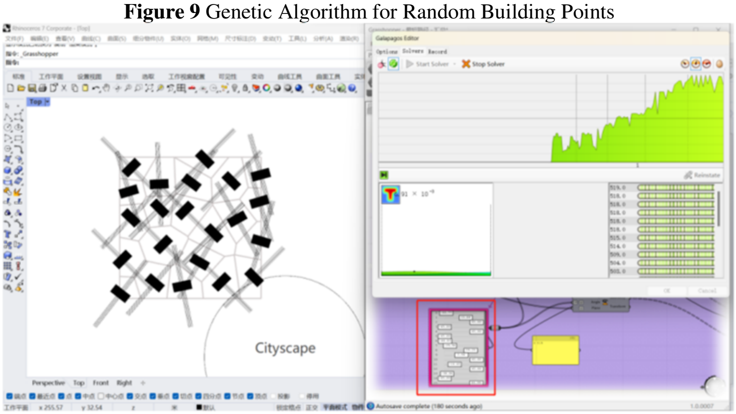


Figure 10 Simulated Annealing Algorithm for Random Building Points

The combined evaluation metrics of the number of buildings, building views, and total distance from buildings to the entrance are 0.643 and 0.677 for genetic algorithm and simulated annealing algorithm respectively (random seed of the Populate Geometry component equals 95). Although the score for simulated annealing algorithm is slightly higher, its computation time is relatively long. Moreover, the optimal solution generated is essentially a fuzzy, approximate range. Therefore, for view deviation, the genetic algorithm is selected as the optimal strategy (Figures 11 and 12).

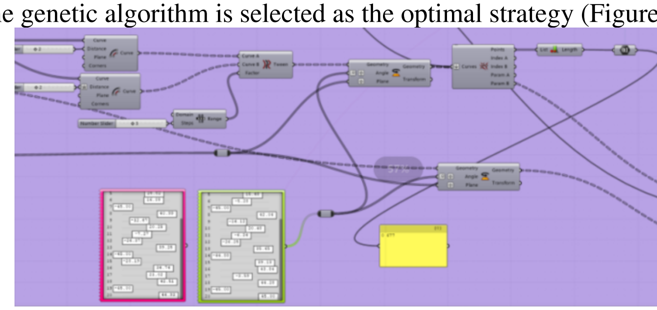


Figure 11 Result of Genetic Algorithm

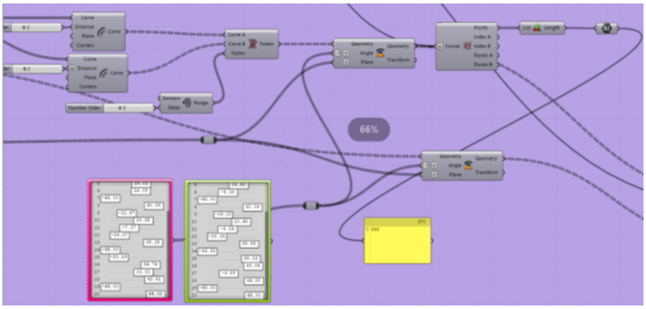


Figure 12 Result of Simulated Annealing Algorithm

2.5 Assessment of Building View Deviation

Among the factors influencing residential orientation, building views are the dominant determinants in building layout, provided that approximate north-south alignment and sunlight distances between buildings can be ensured. Residential orientation is often not determined by the optimal sunlight direction, but a balance between sunlight, other landscape, and view requirements (Figure 12).

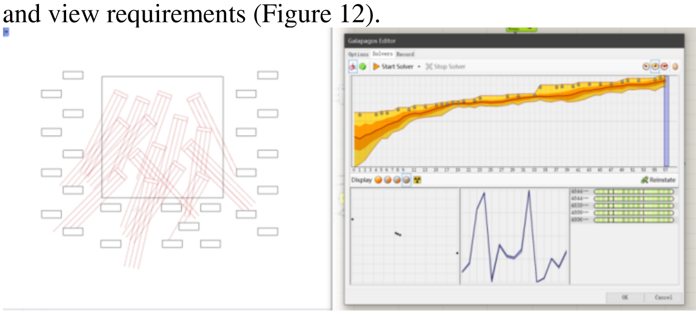


Figure 13 Result of Simulated Annealing Algorithm

In this generative experiment, surrounding buildings obstructed site views. The optimal angle for building deviation, considering maximum visible distance, ranged from 45° southeast to 45° southwest. Facade widths were sampled, and distances to nearest obstacles directly in front were overlaid and weighted. This value, combined with building views and count, assessed random building points from Populate Geometry (Figure 13). Larger values indicated better views. Genetic algorithms then determined optimal deviation angles for each building, achieving the best combination for the entire site.

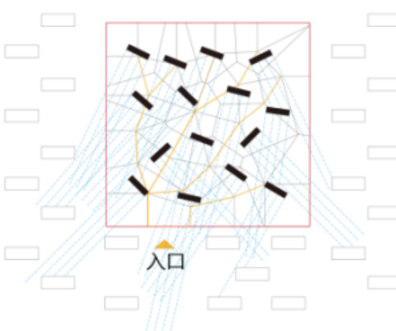


Figure 14 Result of Simulated Annealing Algorithm

3. Conclusion

In this paper, we propose adaptive optimization strategies for residential building design issues using selected algorithms, namely the Wool Algorithm, Slime Mold Algorithm, Dijkstra's Algorithm, Simulated Annealing Algorithm, and Genetic Algorithm. We obtain uniformly distributed building locations by constraining random points and regulating building positions. Voronoi tessellation improves spatial distribution based on individual unit characteristics. The optimization algorithm analyzes site structure, revealing planning typologies and spatial hierarchy divisions. Building orientation angles are determined for optimal solutions. Our study provides insights into design driven by optimization algorithms in residential areas. Performance comparison of five algorithms identifies Dijkstra's and Genetic Algorithm as optimal for path optimization and visual perspective adjustment in residential design.

References

[1] Zhang Baizhou; Li Biao; Exploration on Architectural Space Layout Based on Multi-agent and Dijkstra Algorithm——A Case of the Residence Generative Design. *Urbanism and Architecture* 2020,17(27):7-10+20

[2] Research on the influence of topographic regularity in the automatic layout of industrial parks based on genetic algorithms. *Digital technology - the whole building life cycle——Proceedings of the 2018 National Symposium on Teaching and Research on Digital Technology in Architecture for Architecture Faculties*.,2018:98-103.

[3] Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. science, 220(4598), 671-680.

[4] Van Laarhoven, P. J., & Aarts, E. H. (1987). Simulated annealing. In Simulated annealing: Theory and applications (pp. 7-15). Springer, Dordrecht.

[5] Abramson, D. (1991). Constructing school timetables using simulated annealing: sequential and parallel algorithms. Management science, 37(1), 98-113.

[6] Tracey, N., Clark, J., & Mander, K. (1998). Automated program flaw finding using simulated annealing. *ACM SIGSOFT Software Engineering Notes*, 23(2), 73-81.

[7] Ekren, O., & Ekren, B. Y. (2010). Size optimization of a PV/wind hybrid energy conversion system with battery storage using simulated annealing. *Applied energy*, 87(2), 592-598.

[8] Hong, W. C. (2011). Traffic flow forecasting by seasonal SVR with chaotic simulated annealing algorithm. *Neurocomputing,* 74(12-13), 2096-2107.

[9] Sonmez, F. O. (2008). Optimum design of composite structures with simulated annealing. *Composite Structures,* 86(1-3), 69-82.

[10] AlHalawani, S., Yang, Y. L., Liu, H., & Mitra, N. J. (2013). Interactive facades analysis and synthesis of semi-regular facades. In Computer Graphics Forum (Vol. 32, No. 2pt3, pp. 215-224).

[11] Yeh, I. C. (2006). Architectural layout optimization using annealed neural network. *Automation in Construction*, 15(4), 531-539.

[12] Michalek, J. J., Choudhary, R., & Papalambros, P. Y. (2002). Architectural layout design optimization. *Engineering optimization*, 34(5), 461-484.

[13] Huang, R. H., Chen, Y. P., Tsai, H. W., & Shen, C. C. (2011). Gbps: a new energy efficient routing protocol for wireless sensor networks. *Journal of Information Hiding and Multimedia Signal Processing*, 2(2), 173-182.

[14] Boryczka, U., & Ciura, G. (2015). Path planning for UAV performing detection using Dijkstra algorithm. *Journal of Theoretical & Applied Mechanics*, 53(2).

[15] Wang, K. C., & Chi, Y. C. (1998). A revised Dijkstra's algorithm for the path planning of a cooperative carrying robot system. In Proceedings of IEEE International Conference on Robotics and Automation (Vol. 3, pp. 2611-2616).

[16] Krajnik, T., Nitsche, M., Pedre, S., Preucil, L., & Mejail, M. (2018). The best of both worlds: Making robot navigation in virtual environments more efficient by leveraging state information. In 2018 European Conference on Mobile Robots (ECMR) (pp. 1-6).