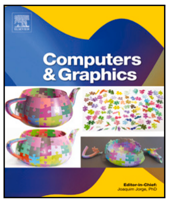
Computers & Graphics 127 (2025) 104175

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag



Technical Section

CVTLayout: Automated generation of mid-scale commercial space layout via Centroidal Voronoi Tessellation[☆]

Yuntao Wang, Wenming Wu[*], Yue Fei, Liping Zheng[*]

School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230601, China

ARTICLE INFO

Keywords:
Commercial space layout
Design optimization
Hierarchical structure
Voronoi diagram

ABSTRACT

The layout of commercial space is crucial for enhancing user experience and creating business value. However, designing the layout of a mid-scale commercial space remains challenging due to the need to balance rationality, functionality, and safety. In this paper, we propose a novel method that utilizes the Centroidal Voronoi Tessellation (CVT) to generate commercial space layouts automatically. Our method is a multi-level spatial division framework, where at each level, we create and optimize Voronoi diagrams to accommodate complex multi-scale boundaries. We achieve spatial division at different levels by combining the standard Voronoi diagrams with the rectangular Voronoi diagrams. Our method also leverages Voronoi diagrams' generation controllability and division diversity, offering customized control and diversity generation that previous methods struggled to provide. Extensive experiments and comparisons show that our method offers an automated and efficient solution for generating high-quality commercial space layouts.

1. Introduction

Layout design is one of the most common tasks in the field of immersive computer graphics, such as computer games, virtual reality, and animated movies [1–5]. In mid-scale commercial architectural design, such as shopping malls, a well-designed commercial environment can significantly enhance customers' understanding of direction and navigation capabilities, prevent the appearance of desolation in stores, and assist customers in locating the products they need. This can greatly improve the shopping experience and prolong the duration of shopping visits. In addition, it is imperative to consider fire safety and evacuation requirements to ensure the safety and efficient evacuation of people in emergencies. Therefore, the computational layout design targeted at commercial spaces has significant research value and importance.

We use the term "mid-scale" [6] to refer to scenarios such as shopping malls, which are larger than small-scale environments (e.g., homes) but smaller and more detailed than large-scale spaces (e.g., cities). Significant efforts have been directed toward small-scale layouts [7–9] and large-scale layouts [1,10,11]. Small-scale layouts typically focus on creating comfortable environments for families within a limited number of rooms, particularly through furniture arrangement. Large-scale layouts, on the other hand, prioritize broader structural elements like zoning, with less emphasis on unit-level details. In contrast, mid-scale commercial spaces require a unique balance between global scale and local functionality, demanding detailed optimization of subregions. The design of mid-scale commercial space layouts involves several challenges. (1) Commercial spaces consist of numerous units with specific configurations, and vary significantly due to differences in business types, target audiences, and specific functional requirements. This diversity presents challenges in defining a universal set of rules or guidelines for automation. (2) Commercial spaces involve complex design considerations, such as branding, space occupancy, and crowd evacuation capability. These factors require a profound understanding of the principles of commercial interior design and their incorporation into the automation process. (3) The lack of suitable datasets specifically tailored for commercial spaces poses a significant barrier to the application of deep learning methods. These methods rely heavily on large-scale annotated training data to learn patterns and generate accurate outputs, which limits the application of deep learning models in this field.

Designing commercial space layouts through trial and error has become inefficient and impractical, as it is time-consuming and difficult to adapt to the rapidly changing needs of commercial spaces and consumer behavior. Peng et al. [12] employ deformable templates-based segmentation to divide regions in the layouts. However, the need for predesigned templates leads to a relatively rigid layout. Wu et al. [13] perform well when dealing with orthogonal boundaries, but when faced

☆ This article was recommended for publication by Y. Tsui Lee.
* Corresponding authors.
E-mail addresses: wyt@mail.hfut.edu.cn (Y. Wang), wwming@hfut.edu.cn (W. Wu), fy@mail.hfut.edu.cn (Y. Fei), zhenglp@hfut.edu.cn (L. Zheng).

https://doi.org/10.1016/j.cag.2025.104175
Received 28 August 2024; Received in revised form 15 January 2025; Accepted 22 January 2025
Available online 31 January 2025

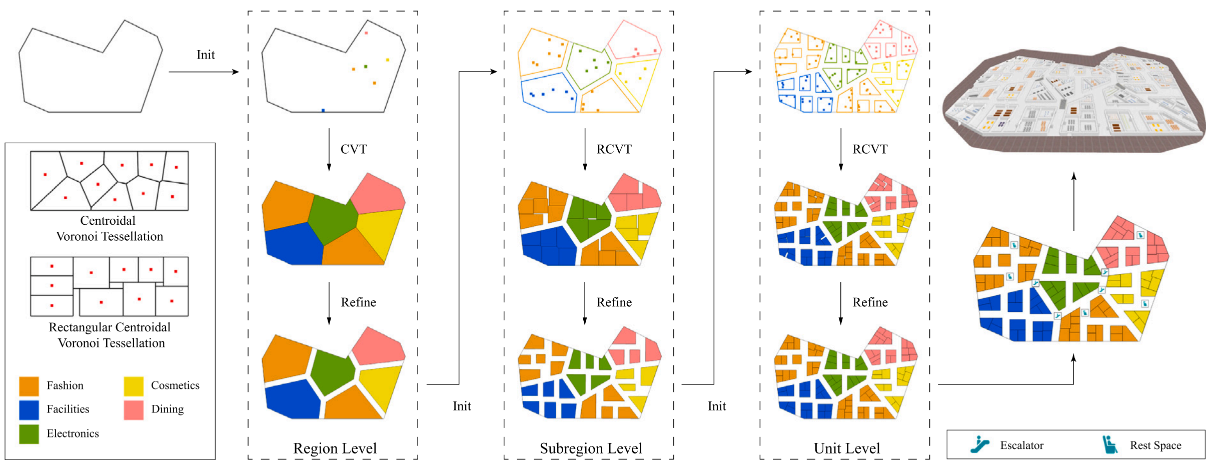


Fig. 1. Overview of our framework. Based on the input boundary, the layout design is achieved through a hierarchical approach. Each dashed box represents a distinct level, with the interior of each box outlining the specific processes associated with that level. The overall process can be summarized as follows: (1) Generate random sites within each area. (2) Optimize sites and generate regions using centroidal optimization. (3) Post-process the generated layout. In the end, escalators and rest areas are added automatically to appropriate locations.

with more complex boundaries, such as those involving slanted edges, extensive additional constraint conditions must be designed in advance. These methods often rely on the construction of the path to adjust the units or require the design of complex constraints, lacking a simple method to exert control by influencing a specific unit. Designers should adhere to specific principles [14] when designing layouts of commercial spaces: (1) Strive for a balanced space allocation and provide adequate space visibility within the overall layout; (2) Provide consumers with quick routes to their destinations; and (3) Increase the variety and flexibility of the space to help customers orient themselves.

In this paper, we focus on the conceptual design of commercial space layouts, introducing a cutting-edge approach leveraging Centroidal Voronoi Tessellation (CVT) for the automated generation of commercial space layouts. By dividing space into distinct regions associated with these sites, Voronoi diagrams help designers efficiently configure buildings and functional areas. This method enables both designers and users to intuitively grasp the attributes and service areas of each region. Moreover, the use of Centroidal Voronoi Tessellation (CVT) ensures a more balanced and equitable division of space. As shown in Fig. 1, we employ a multi-level spatial division framework that allows for a progressive exploration of more detailed designs. At each level, we generate and optimize Voronoi diagrams to accommodate complex, multi-scale boundary constraints. Within this framework, we combine standard Voronoi diagrams with rectangular Voronoi diagrams. The regions in the rectangular Voronoi diagrams align with rectangular boundaries, making them ideal for the regular geometric shapes commonly found in commercial space layouts. Furthermore, by utilizing the boundary-aligned rectangular Voronoi diagram, we ensure that the orientation of the divided areas aligns with the boundaries, enhancing its suitability for layout generation.

Our framework capitalizes on the inherent controllability and diversity of Voronoi diagram generation, enabling users to perform customizable control over the design process. Users can tailor the location, shape, and characteristics of the emerging regions, ensuring that the final layout aligns closely with predefined design objectives and constraints. This level of customization and control distinguishes our approach from previous methods, which often fell short in accommodating the unique demands of commercial space planning. To validate the effectiveness of our design, we employ various evaluations. The results from extensive experiments and comparative analyses underscore the potential of our method to set a new benchmark for automated commercial space layout design. We contribute the following:

- A hierarchical framework for the automated generation of commercial space layouts, which utilizes various Voronoi diagrams and adopts a divide-and-conquer approach to treat different regions independently, enabling more controllable space division.
- A boundary-aligned rectangular Voronoi diagram for generating layouts within complex boundaries and a method to optimize regions by applying centroidal Voronoi tessellation to rectangular Voronoi diagrams enabling a more regular space division.

2. Related work

2.1. Architectural layout design

Automatic generation of interior architectural layouts is one of the typical applications in the field of architectural layout generation. A substantial amount of research has been conducted on small-scale residential design [7–9]. Residential layouts typically aim to create a few rooms, which are usually defined by horizontal or vertical edges, resulting in a relatively simple structure. Additionally, furniture placement plays a crucial role in the design of residential layouts. Liu et al. [15] propose a method for generating floorplans of prefabricated concrete houses by considering functional, design, and manufacturing constraints. Fisher et al. [16] synthesize functional 3D scenes of various sizes by inferring possible human activities. In Laignel et al. [17], interior layouts are generated using grid-based spatial representation and optimization algorithms that incorporate various constraints. Existing methods make significant contributions to the field of controllable scene generation. Yu et al. [18] introduce Clutter Palette, which enhances user control in detailing indoor environments by providing context-aware suggestions for object placement within a scene. Similarly, Zhang et al. [19] improve efficiency in scene editing by enabling users to adjust groups of objects simultaneously, facilitating rapid exploration of various compositional possibilities. In addition, Zhang et al. [20] present an interactive pipeline for manipulating elastic boxes, allowing users to dynamically refine floorplans in real-time. Zhang et al. [21] simplify traditional object selection and arrangement tasks through a data–driven approach that offers intelligent object placement suggestions based on user cursor movements. While these methods highlight the diversity and potential of controllable scene generation, their primary focus has been on small-scale indoor environments, limiting their applicability to larger or more complex scenes.

Various methods have been proposed for large-scale urban design [1,10,11]. Several broader factors, such as population distribution, public transportation, and sunlight, are key considerations in urban design. In this context, the distribution of buildings tends to be more sparse, with less emphasis placed on their individual shapes. Most of the existing methods for urban layout design follow the paradigm of generating street networks first and then subdividing the street-bounded blocks into parcels. Sun et al. [22] introduce a rule-based rewriting method for generating road networks used in urban modeling, while layered road networks are utilized in Galin et al. [23]. Moreover, Aliaga et al. [24] propose a comprehensive algorithm for urban layout synthesis that utilizes attribute intersections and aerial images. Nishida et al. [25] synthesize road networks based on instances. Yang et al. [26] introduce a streamline-based and template-based method to divide the regions. Then, deforming templates are achieved in the method proposed by Peng et al. [12]. Geometric generation is also a crucial aspect of design, and Vanegas et al. [27] develop an interactive system that utilizes geometric and behavioral modeling for urban space design, including generating geometrically and functionally reasonable parcels. Space Plan Generator (SPG) is introduced by Das et al. [28], which automates architectural design through retrieval and decomposition using K-d trees. Hua et al. [10] employ integer programming for urban design, focusing on factors such as area and sunlight. However, their reliance on regular grids and predefined templates limits flexibility in architectural design. Reinforcement Learning is applied in aha et al. [29] to address urban design problems. Chu et al. [30] utilize a generative model to generate city road layouts. Vanegas et al. [31] employ an interactive approach to design 3D city models. Skeleton-based and OBB-based methods are used for region division in Vanegas et al. [32]. Chen et al. [33] propose a hierarchical approach for co-generating parcels and streets.

In commercial space design, Fahmy et al. [34] complete the design of shopping center by allocating facility locations. Feng et al. [6] generate commercial space layouts based on the pedestrian crowd. Zhang et al. [35] further propose various outdoor commercial space design modes by determining paths based on human movement and completing layout generation. In Zhang and Li [36], a method based on geometric modeling and form encoding is proposed for commercial space layout generation, following the design process of architects.

2.2. Voronoi-diagram-based layout design

Voronoi diagram is a spatial division method widely used in layout. Architects use the Voronoi diagram to achieve different spatial structures in urban planning, park design, and other areas of space layout Nowak [37]. In Guo et al. [38], the Voronoi diagram is employed to optimize the rural community layouts. Al-Dahhan and Schmidt [39], Chi et al. [40], Huang et al. [41] utilize the Voronoi diagram to divide regions for robot path planning in various environments, while Tahilyani et al. [42] employ the method for division and assisting in path planning for intelligent vehicles. Wang et al. [43] apply it to bus stop distribution and road network optimization. Yao et al. [44] propose a method using the approach to adjust the population distribution in market areas based on density. Additionally, Lei et al. [45] utilize the Voronoi diagram to generate lattice porous structures, making the structure controllable through the distribution of sites. Similarly, Vavilova et al. [46] study the spatial distribution of galaxies using the Voronoi diagram in astronomical research. Additionally, the Voronoi Adaptive Clustering method is applied in Ma et al. [47] to optimize the position of nodes in wireless sensor networks.

Centroidal Voronoi Tessellation (CVT) is derived from the Voronoi diagram and provides a uniform division of regions. Richter et al. [48] use CVT to optimize waste management areas. A surface layout can also be achieved by employing CVT in Larrea et al. [49]. Abedi and Wichman [50,51] employ CVT to optimize the placement of base stations in cellular networks. In Yadav et al. [52], a structural design method is implemented by incorporating stress-weighted factors into CVT. D'Acunto et al. [53] use CVT to monitor mobile sensors in marine environments. CVT is also applicable in the field of data distribution.

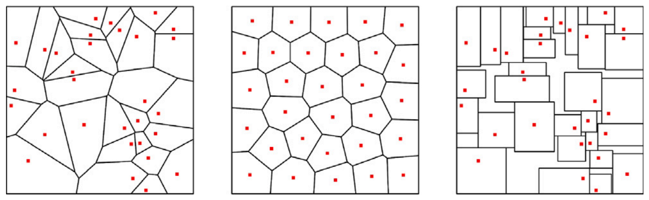


Fig. 2. Voronoi diagrams. Left: Voronoi diagram. Middle: CVT. Right: Rectangular Voronoi diagram.

In the work of Wu et al. [54], multi-level Voronoi tessellation is used to address visualization challenges in clustered graphs, particularly for complex relational network graphs. However, the standard Voronoi diagram employed in this study does not meet the specific shape requirements essential for architectural layouts. The combination of CVT with Conditional Generative Adversarial Networks is used in Chen et al. [55] to obtain uniformly distributed samples for generating virtual samples. In Wang et al. [56], the regular CVT method is used to divide regions. Xin et al. [57] propose a computation method based on capacity constraints.

Choi and Kyung [58] propose to calculate the rectangular Voronoi diagram, which involves division within rectangular boundaries. Chatzikonstantinou [59] apply the rectangular Voronoi diagram to generate small-scale individual buildings and introduce dimension constraint rectangular Voronoi diagram for the automatic generation of 3D layouts. However, it still faces challenges in handling complex boundaries, and its complex optimization process makes it difficult to apply in large-scale layouts. A similar method called the orthogonal Voronoi diagram is proposed in Wang et al. [60], which can generate orthogonal regions, but it may result in the formation of narrow and elongated shapes.

3. Preliminary

3.1. Centroidal Voronoi Tessellation

Voronoi diagram is a widely employed geometric structure that divides space into multiple cells based on a given set of sites $S = \{s_i\}$. As depicted in Fig. 2(a), Each cell $V = \{v_i\}$, represented as a polygon, corresponds to a specific s_i and encompasses it, ensuring that any point within the cell is closer to its corresponding s_i than to any other site in the set. The definition of v_i is as follows:

$$v_i = \{s \mid \|s - s_i\| \le \|s - s_j\|, \forall j \ne i\} \tag{1}$$

In the equation, the symbol $\|.\|$ represents the Euclidean distance. With the Voronoi diagram, Centroidal Voronoi Tessellation (CVT) [61] refers to a specific type of Voronoi tessellation where each site s_i is positioned at the mass centroid s_i^* of the corresponding cell v_i, as follows:

$$s_i = s_i^* = \frac{\int_{v_i} s\rho(s)ds}{\int_{v_i} \rho(s)ds} \tag{2}$$

$\rho(s)$ denotes the density at s within the given domain. Fig. 2(b) shows an example of CVT. To obtain CVT, a common method is to iteratively move each site to the center of the corresponding cell until convergence is reached [62].

3.2. Rectangular Voronoi diagram

The rectangular Voronoi diagram is a special type of Voronoi diagram, where cells are generated by constructing rectangular regions corresponding to each site [58]. Fig. 2(c) shows an example of the rectangular Voronoi diagram. For a site s_i, the other sites are divided into four sets (up, bottom, left, and right) according to relative direction, as shown in Fig. 3(a). The rectangular region is obtained by calculating

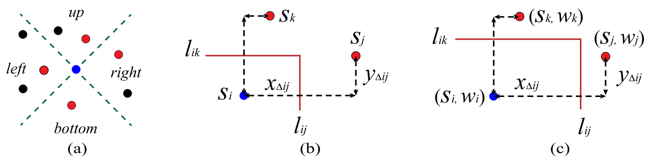


Fig. 3. Generation of Rectangular Voronoi diagram. (a) Sites are divided into four sets (up, bottom, left, and right). (b) Generation of midlines between two sites in the rectangular Voronoi diagram. (c) Generation of midlines between two sites in the weighted rectangular Voronoi diagram.

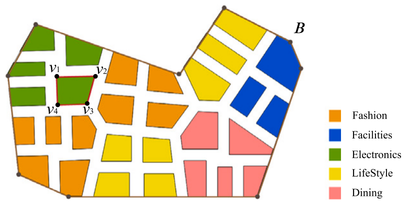


Fig. 4. Layout representation. The commercial space layout can be represented as a set of non-overlapping polygonal areas.

four midlines between the current site s_i and the closest site s_j in all four sets. The distance between s_i and s_j is calculated using $x_{\Delta ij}$ or $y_{\Delta ij}$ according to the relative direction. Fig. 3(b) shows the generation of the midline l_{ij} between s_i and s_j, and it is determined as follows:

$$l_{ij} : \begin{cases} x = \frac{x_i+x_j}{2}, & x_{\Delta ij} \geq y_{\Delta ij} \\ y = \frac{y_i+y_j}{2}, & x_{\Delta ij} < y_{\Delta ij} \end{cases} \quad (3)$$

Here $s_i = (x_i, y_i)$, $s_j = (x_j, y_j)$, $x_{\Delta ij} = \left|x_i - x_j\right|$, $y_{\Delta ij} = \left|y_i - y_j\right|$. If a boundary edge in the direction is closer to s_i than the current l_{ij}, then the edge will be considered as the new value of l_{ij}. Due to the strict constraints on the direction of the edges, there may be gaps between cells.

3.3. Weighted rectangular Voronoi diagram

Weight can be introduced when calculating the rectangular Voronoi diagram. We combine weight into the rectangular Voronoi diagram for a constrained generation. Each site s_i has a weight w_i. The method for calculating the midline l_{ij} with weight between adjacent sites s_i and s_j in each of the four sets is as follows:

$$l_{ij} : \begin{cases} x = x_i - \frac{w_i}{w_i+w_j} x_{\Delta ij}, & x_{\Delta ij} \geq y_{\Delta ij},\ x_i \geq x_j \\ x = x_i + \frac{w_i}{w_i+w_j} x_{\Delta ij}, & x_{\Delta ij} \geq y_{\Delta ij},\ x_i < x_j \\ y = y_i - \frac{w_i}{w_i+w_j} y_{\Delta ij}, & x_{\Delta ij} < y_{\Delta ij},\ y_i \geq y_j \\ y = y_i + \frac{w_i}{w_i+w_j} y_{\Delta ij}, & x_{\Delta ij} < y_{\Delta ij},\ y_i < y_j \end{cases} \quad (4)$$

Here w_i represents the weight of s_i. In general, the higher the weight, the larger the cell for fixed sites.

4. Overview

An overview of our framework to generate commercial space layouts is shown in Fig. 1.

Problem. Given an input layout boundary B, such as the boundary of a shopping mall, our goal is to generate the layout G within it and allocate the space to each region.

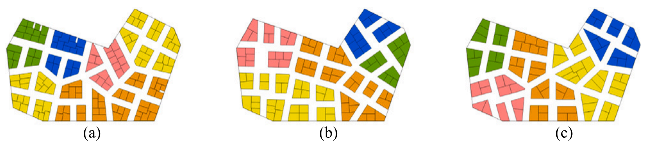


Fig. 5. The results generated by setting different a_s at the final level. (a) 50 m^2. (b) 64 m^2. (c) 80 m^2.

Layout representation. As shown in Fig. 4, we take the input of a boundary B represented as a polygon. The commercial space layout can be represented as a set of areas $G = \{R_i\}$, where $R_i = \{P_i, t\}$ are non-overlapping areas. We use the polygon $P_i = \{v_i\}$ to represent the shape of the area, and t indicates the function of the area, such as electronics, fashion, and dining. The areas within the boundary B that remain unoccupied are designated as paths, primarily used for separating units. Entrances can be formed at the intersection of the paths and the boundary B, allowing agents to enter and exit the layout.

Methodology. We propose a hierarchical layout generation framework (Fig. 1). Specifically, we take a set of regions from the previous level as input, generate their respective sub-regions, and use these sub-regions as input for the next level. Each level can be divided into three steps: (1) Generate random sites within each area. (2) Optimize the sites and achieve a more suitable division through centroidal optimization. (3) Post-process the generated layout based on specific requirements, such as creating accessible paths or filling gaps between areas. Depending on the specific level, we can choose to use a standard Voronoi diagram or a rectangular Voronoi diagram for spatial division. In the initial phase, when the space to be divided is relatively large, using the standard Voronoi diagram in the first level allows for increased flexibility and unique arrangement. However, as the size of the region decreases, excessive variations in path directions of standard Voronoi diagrams can become confusing for visitors. Rectangular Voronoi diagrams offer better control over the directionality and configuration of the regions, ensuring a more intuitive and realistic layout.

5. Method

Our method divides the layout into multiple levels, achieving spatial division through optimization at each level. Additionally, constraints can be added to meet specific layout requirements. Our layout design framework consists of two kinds of Voronoi diagrams: the standard Voronoi diagram and the rectangular Voronoi diagram. Centroidal Voronoi Tessellation (CVT) can be used to optimize the standard Voronoi diagram. To further optimize the rectangular Voronoi diagram, we propose a centroidal Voronoi tessellation called RCVT.

For each level of layout design, given the area to be divided, we first calculate the number of sites $n = \frac{a_r}{a_s}$ for the Voronoi diagram based on the total area a_r of the region to be divided and estimated area a_s of the sub-region. a_s affects the number of sites and the area of sub-regions, as shown in Fig. 5. Sites can also be randomly generated within the area according to the user's requirements, allowing for flexibility and customization in the design process.

We first introduce the method for division within a single level, while more information about hierarchical spatial division can be found in Appendix A.

5.1. RCVT-based spatial division

5.1.1. Boundary-aligned RVD

We make specific adjustments to the computation method of the Rectangular Voronoi Diagram (RVD) to ensure that the orientations of the divided regions align with the boundaries and paths, even when dealing with complexities such as sloped or skewed edges in

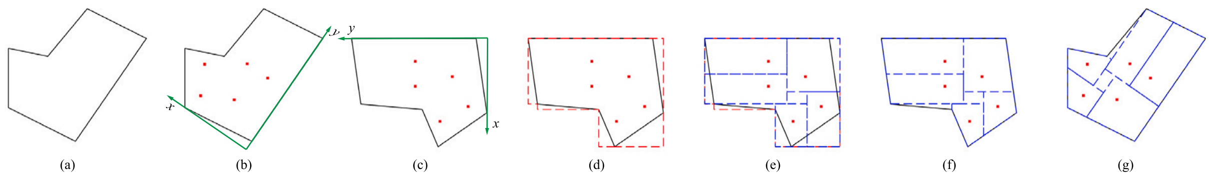


Fig. 6. The Computation of Boundary-aligned RVD. (a) Input boundary. (b) Generate sites. (c) Calculate the primary direction and rotate the coordinate system. The green segments in (b) and (c) represent the primary direction of the region. (d) Approximate the boundary using orthogonal continuous segments. (e) Compute the cells constrained by the approximated boundary. (f) Clip the portions that extend beyond the boundary. (g) Restore the original coordinate system.

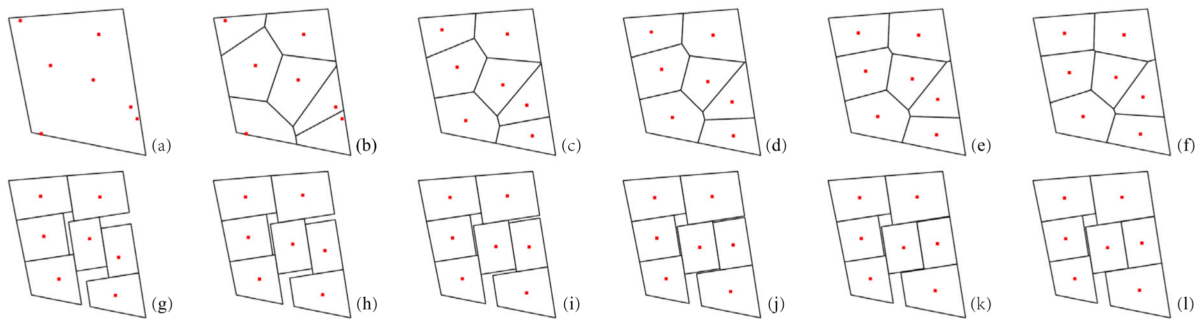


Fig. 7. The Computation of RCVT. The figure illustrates the iterative process of centroid optimization for sites. (a) represents the input boundary, while (b-l) depicts a portion of a continuous optimization process. During stages (b–f), random sites are generated, and CVT is used for optimization. In stages (g-l), the previously optimized sites in (f) are used as the initial configuration, and RCVT is employed for further optimization.

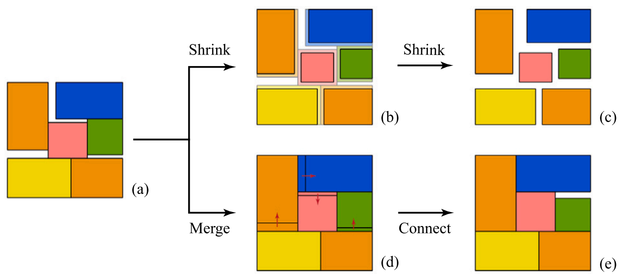


Fig. 8. Layout refinement. To address gaps among cells, (b) and (c) illustrate the process of forming accessible paths by shrinking the boundaries of the sub-regions, with gaps serving as pedestrian-accessible spaces. (d) demonstrates the process of subdividing the gaps and merging them into the surrounding sub-regions. (e) illustrates the process of connecting inaccessible units to accessible paths..

commercial settings. Our method is shown in Fig. 6, beginning with a polygonal boundary as the input. Initially, before performing spatial division, we identify the primary direction by selecting the longest edge of the boundary. We then rotate the boundary to align with this primary direction, as shown in Fig. 6(b–c), which ensures the orientations of the generated paths and input boundaries remain consistent.

For boundaries with sloped sides, a series of connected horizontal and vertical lines approximate the slope, completely enclosing the boundary within a fitted polyline to form a rectangular representation. This approach of replacing the original boundary with a fitted polyline composed of orthogonal segments adapts better to shapes (Fig. 6(e)). Algorithm 1 outlines the computational process, offering a refined solution that better accommodates the complexities of commercial space layouts by ensuring alignment with architectural boundaries and facilitating more navigable and visually coherent spaces.

Algorithm 1 Boundary-aligned RVD Computation

```
Input: Sites $S = \{s_i\}_1^n$   Boundary $B = \{v_i\}$
Output: Cells $C = \{c_i\}_1^n$
 1: $direction$ ← calc primary direction ($B$)
 2: rotate coordinate ($direction$)
 3: $A$ ← approximate boundary ($B$)
 4: for $i = 1, \dots, n$ do
 5:     $n\_sites$ ← find neighbor sites ($s_i$, $S$);
 6:     $n\_edges$ ← find neighbor edges ($s_i$, $A$);
 7:     $up$ ← calc up edge ($s_i$, $n\_sites_u$, $n\_edges_u$)
 8:     $bottom$ ← calc bottom edge ($s_i$, $n\_sites_b$, $n\_edges_b$)
 9:     $left$ ← calc left edge ($s_i$, $n\_sites_l$, $n\_edges_l$)
10:     $right$ ← calc right edge ($s_i$, $n\_sites_r$, $n\_edges_r$)
11:     $c_i$ = build rect ($up$, $bottom$, $left$, $right$)
12:     $c_i$ = clip ($c_i$, $B$)
13: end for
14: restore coordinate ($direction$)
```

Algorithm 2 RCVT Computation

```
Input: Sites $S = \{s_i\}_1^n$   Boundary $B = \{v_i\}$
Output: Sites $S = \{s_i\}_1^n$   Cells $C = \{c_i\}_1^n$
 1: $S$ ← CVT ($S$, $B$)
 2: $D = \{d_i\}_1^n$
 3: while max $D > threshold$ do
 4:     $C$ ← boundary-aligned RVD ($S$, $B$)
 5:     for $i = 1, \dots, n$ do
 6:         $p_i$ = centroid ($c_i$)
 7:         $d_i = \| p_i - s_i \|$                ▷ Euclidean distance
 8:         $s_i = p_i$
 9:     end for
10: end while
```

5.1.2. RCVT

Once the initialization of specific regions within a level is complete, we obtain a set of sites and their corresponding region divisions, which may exhibit substantial variations in area. To address this issue, we employ Rectangular Centroidal Voronoi Tessellation (RCVT) to refine the

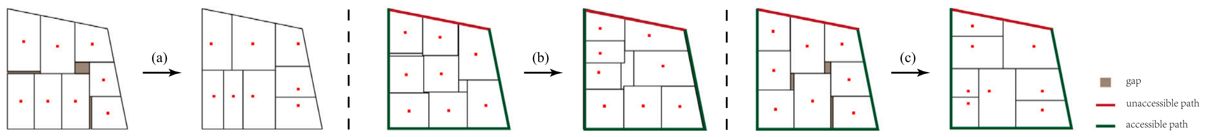


Fig. 9. Layout optimization with simulated annealing. (a) Optimization of space utilization, (b) Optimization of accessibility, and (c) Mixed optimization. For each subplot, the left displays the initial division obtained through RCVT, while the right shows the refined layout after optimization.

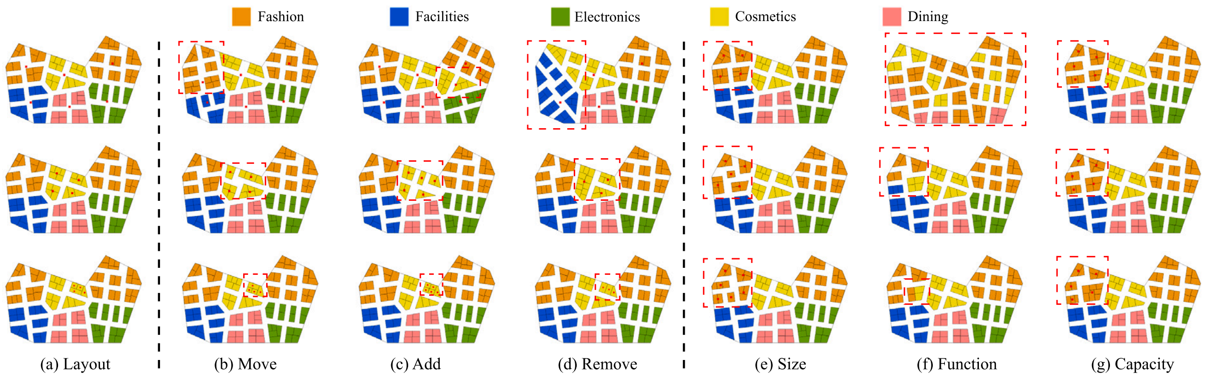


Fig. 10. Site-constrained division. We highlight areas of interest with red frames. (a) illustrates our initial layout, showcasing the sites for *Region*, *Subregion*, and *Unit* from top to bottom. (b–d) display the results obtained through location constraints, achieved by moving, adding, and removing sites in (a) at different levels. (e) demonstrates the generation of four, five, and six *Subregions* within the same *Region* based on the specified quantity. (f) provides constraints on the total number of functions, specified *Subregion* functions, and specified *Unit* functions. (g) demonstrates the process of adjusting the capacity by modifying the weights of specified *Subregion*.

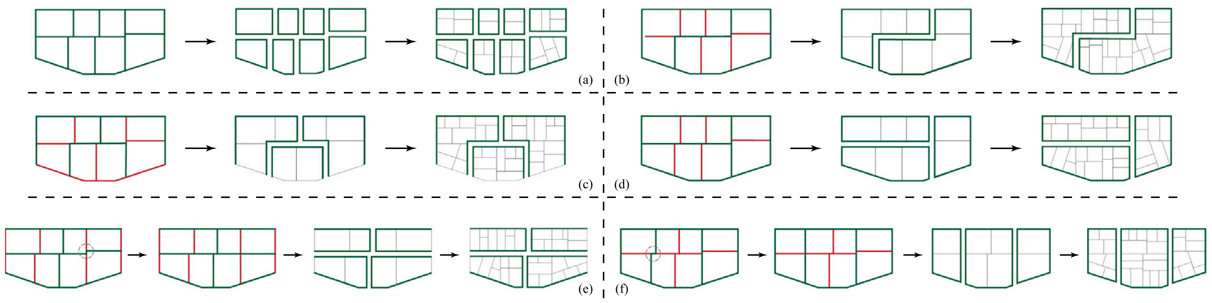


Fig. 11. Path-constrained division: Green segments represent accessible paths, while red segments indicate inaccessible paths. Each group of results in (a–d) is presented in three stages: the left shows the spatial division of a region achieved through RCVT, the middle displays the accessible paths with their actual widths, finalizing the sub-region division and indicating which edges of each sub-region are accessible or inaccessible, and the right presents the spatial division of the *Subregions*, ultimately leading to the formation of the final *Units*. his process ensures that each unit is connected to at least one accessible path, enabling functional and user-friendly layouts. The gray circles in (e) and (f) highlight areas where kinks exist in the accessible paths, which we aligned to create straight paths.

layout. Since boundary-aligned Rectangular Voronoi Diagrams (RVDs) impose strict constraints on each cell's boundaries, directly optimizing the cell centroids could lead to the excessive elongation of neighboring regions. To prevent this, we first perform a set number of iterations using Centroidal Voronoi Tessellation (CVT). Subsequently, we utilize the boundary and sites as input for boundary-aligned RVD and conduct iterative optimization. In each iteration, we calculate the centroid of each cell and reposition the corresponding site to that centroid. The updated sites are then used in the boundary-aligned RVD to compute a new subregion division. The calculation process of RCVT is outlined in the algorithm 2. By optimizing the sites, the entire space will be divided into multiple relatively evenly distributed sub-regions. Fig. 7 illustrates the optimization process.

5.2. Post-processing

We propose two kinds of post-processing: layout refinement and layout optimization.

Layout refinement. After applying RCVT for space division, gaps may appear between cells due to the imposed shape constraints. To address these gaps, we employ two strategies, as illustrated in Fig. 8. For non-final levels, the resulting empty spaces are designated as public areas accessible to pedestrians, as shown in Fig. 8(a–b–c). We treat the boundaries of each unit as accessible paths and shrink them by a fixed width to create walkable pathways. This approach ensures efficient space utilization and maintains pedestrian accessibility throughout the layout. At the final level, gaps are subdivided into smaller rectangles. Subsequently, we traverse these subdivided rectangles and integrate them into adjacent units to minimize the difference between the convex hull area of the merged unit and its actual area, as depicted in Fig. 8(d). For inaccessible units, we create new pathways that connect to existing accessible paths, as depicted in Fig. 8(e).

Layout optimization. We introduce a method based on the simulated annealing algorithm to optimize the divisions obtained from RCVT. This approach aims to minimize gaps and effectively manage the creation of accessible paths, thereby providing better control. Divisions are closely associated with sites. We achieve optimized divisions by adjusting the locations of the sites, thereby affecting the positions, polygon shapes, and topological structures of the corresponding units to meet the optimization objectives. As shown in Fig. 9, we provide approaches for optimizing space utilization, accessibility, and their combination. More detailed information about the simulated annealing algorithm can be found in Appendix B.

5.3. Constrained division

To provide flexibility and customization, ensuring the system meets diverse design requirements, we offer two methods for controlling the division: site-constrained division, which allows users to influence layout division by controlling the location, size, function, and capacity of specific sites; and path-constrained division, which enables users to manage the generation of accessible paths to affect the overall layout structure.

Site-constrained division. The unique property of Voronoi diagrams, where each site is associated with a distinct region, enables intuitive layout control through site manipulation. By applying constraints on location, size, function, and capacity to the sites, we can effectively generate the corresponding layouts, as illustrated in Fig. 10. A comprehensive description of our site-based division method is provided in Appendix C.

Path-constrained division. In addition to site constraints, path management plays a crucial role in shaping the layout. By customizing path configurations, we can control the division of regions, which subsequently influences the spatial division at deeper levels. To achieve this, we employ a path-constrained division process using the simulated annealing algorithm for post-processing. Fig. 11 illustrates the impact of path-based constraints on a specific region. By default, the internal edges of all sub-regions are designated as accessible paths, with their widths adjusted to practical dimensions, as shown in Fig. 11(a). Additionally, alternative path configurations can be applied, as demonstrated in Fig. 11(b–f), where the units resulting from sub-region division are guaranteed to be adjacent to at least one accessible path. If the accessible paths contain short segments that create kinks, as indicated by the gray circles in Fig. 11(e–f), we remove these short segments and align adjacent segments, thereby creating straight paths. This approach provides greater flexibility and adaptability in layout design, enabling the creation of functional, accessible, and user-friendly spaces that align with specific design goals and user needs.

6. Experiment

6.1. Implementation

We use C++ and CGAL [63] to implement our method on a CPU of AMD Ryzen 7 3700X 8-core (3.60 GHz). On average, without considering concurrency, it takes about 20 s to generate each layout. In the generated layout, we use 1 pixel to represent 0.2 m, and in practical applications, paths at different levels can be set with different pixel widths: 7, 5, and 2.5 m. The estimated area a_s used for calculating the number of sub-regions at different levels is 2000 m^2, 400 m^2, 50 m^2. We use a distance of 2 meters as the termination condition in centroid optimization. We test our method to generate layouts on the 11 input boundaries from real-world commercial spaces. To evaluate the generated layouts using space syntax analysis, we employ the open-source space syntax software DepthmapX-0.80 [64], which is commonly used by architects.

6.2. Evaluation

We design the indicators of *Crowd Evacuation* and *Allocation Uniformity* to evaluate the generated layouts, then evaluate the visibility of the layout through the spatial syntax analysis, which is utilized for measuring spatial structures and relationships [65]. It is commonly applied in analyzing large and medium-scale layouts, especially for layouts involving complex spatial elements, such as commercial architecture, urban planning, and exhibition design [66,67]. Visual Graphic Analysis (VGA) is a widely employed method for space syntax analysis. It involves the grid-based representation of a layout, where each grid cell corresponds to a spatial unit. By analyzing the relationships between these grid cells, VGA enables the assessment of spatial structure. We employ VGA to assess the generated layout, resulting in three metrics: *Visual Connectivity*, *Visual Integration*, and *Intelligence*. These metrics help us assess the spatial characteristics and qualities of the layout.

Crowd evacuation. It is a crucial metric of visitor safety in commercial environments. Given the significant influx of people drawn to commercial establishments, it is crucial to prioritize effective crowd evacuation design and planning. The assessment of layout effectiveness in crowd evacuation is a fundamental indicator for evaluating the efficacy of such designs. We use a grid to represent the layout, and a set of entrances $E = \{e_i\}_1^m$ is randomly generated to ensure equal distribution throughout the design. Due to the divergent outcomes in entrance placement resulting from different methods, we opt to readjust the entrances to the closest branch paths that connect to the external areas to ensure the similarity of the entrance. Then, we generate 5000 points $V = \{v_i\}_1^n$ randomly along the layout paths to represent visitors in the grid. These points emulate the movement of visitors, allowing them to transition from their current grid cell to adjacent grid cells, signifying one step. The minimum number of steps required for each visitor to reach any entrance from their initial position is calculated, and the average value of the minimum step counts across all visitors is used as an evaluation metric for the overall layout. A lower step count indicates smoother and faster evacuation, indicative of a higher level of effectiveness in the evacuation design. We calculate the average value of all crowd simulation points as the final metric. The following formula is used to calculate *Crowd Evacuation*:

$$CrowdEvacation = \frac{\sum_{i=1}^{n} \min \|v_i,\ e_j\|,\ e_j \in E}{n} \tag{5}$$

Here$\|.\|$ calculates the shortest steps from the point v_i to the entrance e_j.

Allocation uniformity. It represents the uniformity of *Units* distribution in the layout, indicating the allocation balance of space. Using the area of all *Units* $A = \{a_i\}_1^n$ in the layout, we first obtain the mean value μ of A, and then calculate the degree of difference of *Units* occupying the space using the coefficient of variation. A smaller value indicates a more balanced division. The following formula is used to calculate *Allocation Uniformity*:

$$AllocationUniformity = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(a_i - \mu)^2}}{\mu} \tag{6}$$

Space utilization. Using the area of all *Units* $A = \{a_i\}_1^n$ in the layout, we obtain Space Utilization by calculating the ratio of their sum to the total area of the layout a_l. The following formula is used to calculate *Space Utilization*:

$$SpaceUtilization = \frac{\sum_{i=1}^{n} a_i}{a_l} \tag{7}$$

Visual connectivity. It refers to the count of grid cells that are directly visible from a specific space in grid-based spatial analysis [68]. It quantifies the number of visually adjacent spaces within the line of sight, thereby indicating a space's local visibility characteristics. A higher value of Visual Connectivity indicates better visibility for the given space. We select the average value as a metric to measure the overall layout. We take a layout as an example to demonstrate the visualization of this metric, as shown in Fig. 12(b).

Visual integration. It is calculated based on the number of visual turns required to see a specific space from any other space in the layout [69]. It reflects a space's global visibility characteristics. In essence, a higher value of Visual Integration indicates that the element is more likely to capture visual attention, meaning that people are more likely to see this space. We employ the average value as a quantitative measure to evaluate the overall layout. Fig. 12(c) shows the visualization.

Table 1
Comparison with baseline methods, including Random Orthogonal Partition (ROP), Public-Business Space Generation (PBS), and MIQP Method for Layout Design (MIQP). The bold font means the best results.

Method	*Crowd Evacuation* ↓	*Allocation Uniformity* ↓	*Space Utilization* ↑	*Visual Connectivity* ↑	*Visual Integration* ↑	*Intelligibility* ↑
ROP	15.834	0.332	0.522	299.761	5.267	0.676
PBS [36]	14.852	0.797	0.460	262.163	5.819	0.608
MIQP [13]	15.292	0.479	**0.558**	272.555	5.424	0.719
Ours	**14.702**	**0.246**	0.557	**304.767**	**6.133**	**0.767**

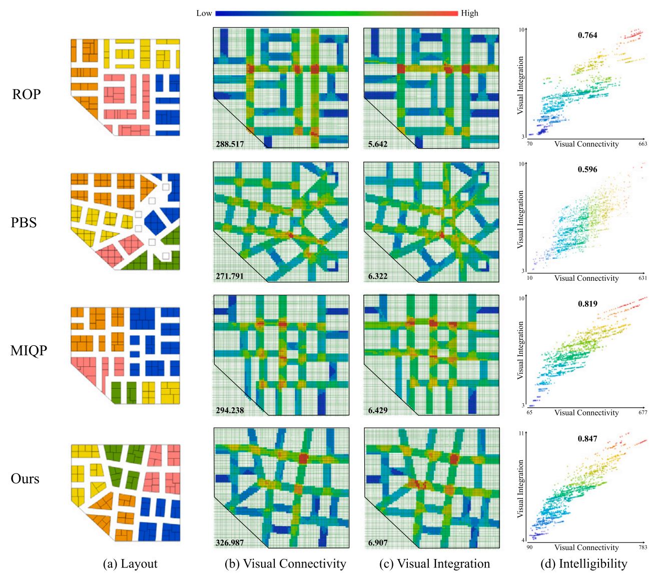


Fig. 12. Visualizaiton of *Visual Connectivity*, *Visual Integration* and *Intelligibility*. Colors from blue to red indicate values from low to high.

Intelligibility. It represents the comprehensibility or understandability of a layout, specifically regarding the ability of visitors to perceive the global layout from a local space within the design [70]. A higher value of Intelligibility indicates that the overall layout structure is easier to understand, which improves navigation and movement convenience. Intelligibility is calculated by measuring the correlation coefficient between Visual Connectivity and Visual Integration. Fig. 12(d) shows the visualization, and the set of points will concentrate on a straight line as the correlation increases.

6.3. Comparison

We compare our method with existing related techniques. To standardize the generated layouts, we uniformly set a fixed width of 9 meters for the main paths. In Table 1, we present the average evaluation metrics calculated across 11 building boundaries that closely resemble real-world scenarios. These metrics provide a comprehensive assessment of the layout's performance under realistic conditions. Our method achieves the best results on metrics of *Crowd Evacuation*, *Allocation Uniformity*, *Visual Connectivity*, *Visual Integration*, and *Intelligibility*, and achieves the second-best in terms of *Space Utilization*. Fig. 13 compares the results generated by these baseline methods and our method.

Random orthogonal partition (ROP). This method generates layouts using simple orthogonal division. For the given layout boundary, the bounding box of each region is obtained, and the box is randomly divided into two parts iteratively until the region's area is below the specified threshold. ROP can only divide regions by generating horizontal or vertical paths randomly. The limited path directions and random division resulted in suboptimal performance for ROP in terms of Crowd Evaluation in Table 1. Furthermore, this may yield some irrational areas. For example, the second and fourth of Fig. 13(a) contain several narrow units. In comparison, our approach uses centroid optimization to achieve more uniform space allocation and unit division and achieves high visibility in the overall layout. In contrast to orthogonal paths, the inclusion of slanted paths and balanced regions allows users to reach their destinations quickly. Furthermore, the characteristics of the rectangular Voronoi diagram lead to the generation of open spaces between regions, thereby further enhancing the visibility of the layout as well.

Public-business space generation (PBS). Zhang and Li [36] divides the layout into public spaces and business spaces by refining architectural design rules. It consists of two parts: the generation of public spaces dominated by geometric prototype strategies and the generation of business spaces dominated by morphological coding strategies. Firstly, skeleton lines are obtained from the boundary polygons, and the longest skeleton line is extended. Multiple atriums are manually designed and expanded as public spaces. Multiple entrances are manually designed and connected to the nearest atrium section. The remaining parts serve as business spaces, and if any area becomes too large, it is iteratively divided. Finally, the units are divided by a fixed column grid. This method is more suitable for addressing certain elongated boundaries, as shown in the third of Fig. 13(b). However, for inputs akin to those depicted in the second and fourth results of Fig. 13(b), there exists a pronounced disparity in the division surrounding the atrium relative to other areas, culminating in a diminished Intelligence value in Table 1. The expansion from the atrium results in a wider main pathway, which leads to a decrease in *Space Utilization* in Table 1. The imprecise column grid division leads to suboptimal performance for PBS regarding Allocation Uniformity, as shown in Table 1. The layouts generated by the PBS exhibit a strong dependence on the specific atrium design, and the final stage requires manual merging of the column grid. Consequently, the quality of the ultimate results remains reliant on the professional knowledge and expertise of the designers.

Mixed integer quadratic programming (MIQP). Wu et al. [13] employs polygon parameterization to decompose polygons into smaller polygons. Constraints are established in a manner compatible with MIQP and parameterization. The generation of complex layouts proceeds through iterative computations. We design specific parameters to complete the layout design. Orthogonal paths result in longer evacuation distances, and MIQP generates some uneven unit combinations and dispersed paths. MIQP achieves suboptimal results in Crowd Evaluation in Table 1. In the fifth example of Fig. 13(c), a winding path is generated, which slows the crowd's movement to their intended destination and reduces the visibility of the layout. While confronting complex boundaries, the application of simplistic constraints may engender discordance with boundaries, as shown in the first of Fig. 13(c). In such cases, more complex constraints tailored to the specific boundary conditions are necessitated. The spaces occupied by different regions in the layout vary significantly in size.

6.4. Ablation study

We incorporate boundary direction for the rectangular Voronoi diagram and employ centroidal optimization to achieve more appropriate region division. To illustrate the influence of different components on the overall layout, we design relevant ablation experiments. Firstly,

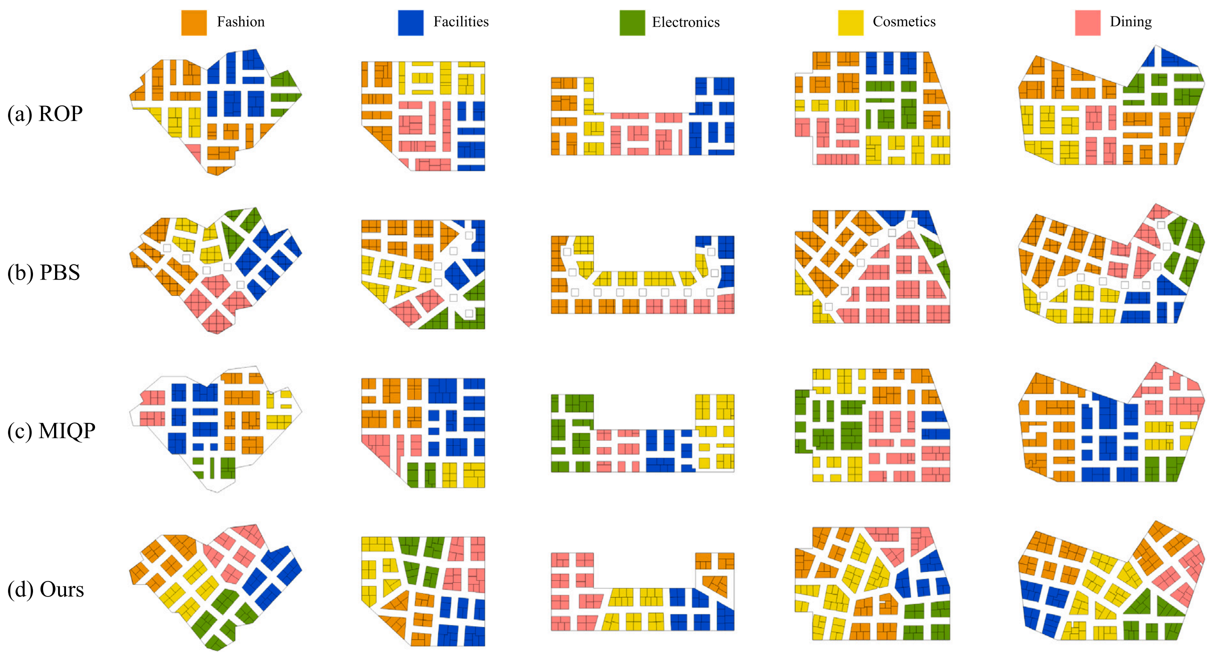


Fig. 13. Comparison with baseline methods. (a) Random Orthogonal Partition (ROP). (b) Public-Business Space Generation (PBS). (c) Mixed Integer Quadratic Programming (MIQP). (d) Our Method.

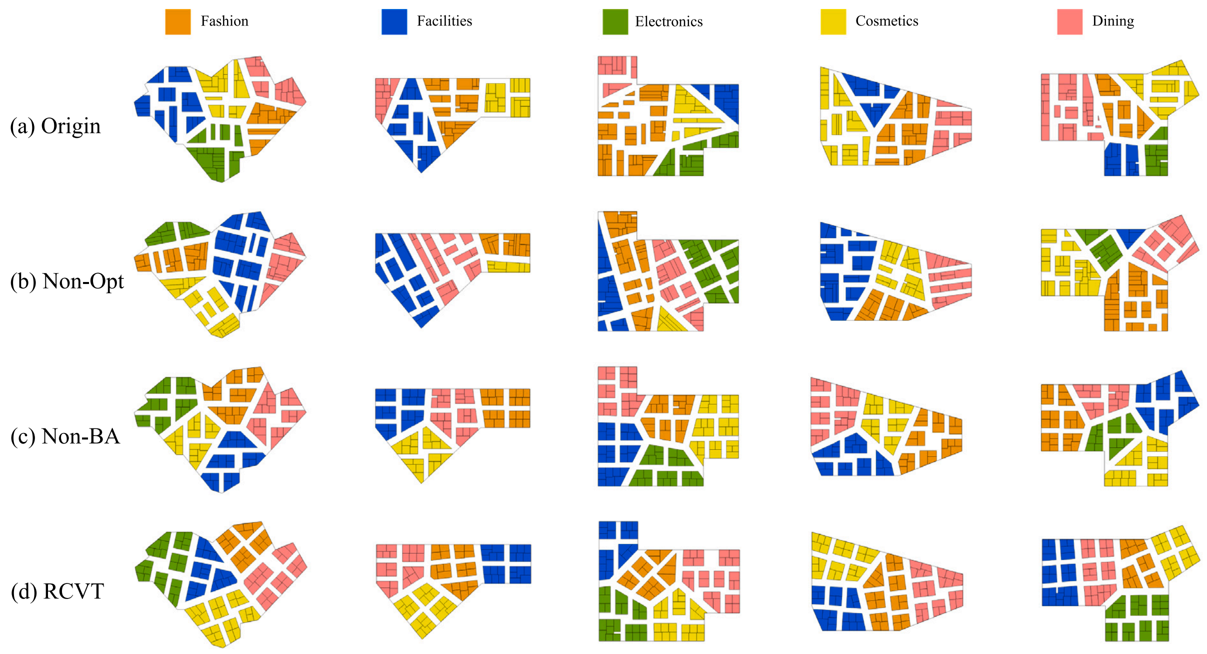


Fig. 14. Ablation. (a–c) show the results obtained by modifying specific aspects of the method. (d) shows the results generated by RCVT.

we generate a layout of sites at random positions, without considering centroid optimization or the alignment of divided areas with boundary orientations, referred to as *Origin*, as illustrated in Fig. 14(a). The region division is highly uneven, with the possibility of some regions having tiny areas and uneven distribution of paths. Additionally, due to the absence of boundary guidance, the directions of the *Units* are unidirectional. Secondly, as shown in Fig. 14(b), we generate layouts without employing centroid optimization, referred to as *Non-Opt*. The distribution of regions in the result is still not ideal, but the directions of the *Units* show a correlation with the boundaries and path directions. Then, we perform region division without aligning the divided areas with boundary orientations, referred to as *Non-BA*, as illustrated in Fig. 14(c). The layout exhibits uniform region division, but the orientations of the units still remain limited to horizontal and vertical directions. As a result, numerous *Units* are not aligned with the direction of external boundaries, and the direction of *Units*

Table 2
Ablation study. AR refers to the Aspect Ratio, where AR_m denotes the mean aspect ratio of all *Units*, and AR_s represents the standard deviation of the aspect ratio. $AR_{>4}$ indicates the number of *Units* with an aspect ratio greater than 4, highlighting the presence of elongated units. AU stands for Allocation Uniformity, with AU_R, AU_S, and AU_U representing the uniformity at the *Region*, *Subregion*, and *Unit* levels, respectively. This metric evaluates how evenly space is distributed at different hierarchical levels. SU refers to Space Utilization, which measures the proportion of utilized space within the layout relative to the total available space. *Non-Opt* refers to the method without centroidal optimization, while *Non-BA* refers to the method without boundary alignment. The bold font highlights the best results for each metric, indicating the most effective method for each aspect of layout design.

Method	$AR_{m\ x\to 1}$	$AR_s \downarrow$	$AR_{>4} \downarrow$	$AU_R \downarrow$	$AU_S \downarrow$	$AU_U \downarrow$	$SU \uparrow$
Origin	2.415	1.794	11	0.433	0.587	0.518	0.570
Non-Opt	2.340	1.518	10	0.447	0.597	0.514	0.583
Non-BA	1.380	0.344	**0**	0.157	0.242	0.240	0.607
RCVT	**1.371**	**0.342**	**0**	**0.155**	**0.226**	**0.231**	**0.612**

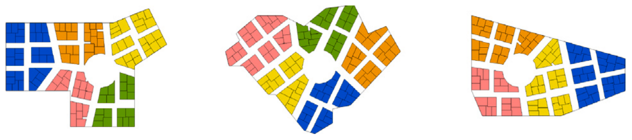


Fig. 15. Design of atrium. Here shows the commercial space layouts generated with an atrium, which is incorporated as a constraint in the generation.

is single and fixed. Finally, we show the results generated by RCVT in Fig. 14(d), which ensures a more balanced region division and adjusts the orientation of areas.

Table 2 presents the quantitative results of ablation studies. We use 11 building boundaries that simulate real-world scenarios as input and compute the average across all layouts to evaluate performance. To evaluate the layout *Units*, we calculate their oriented bounding boxes and determine their aspect ratios (AR). An aspect ratio closer to 1 indicates a more square-like spatial configuration. The *Origin* and *Non-Opt* methods in Table 2 exhibit higher AR_m and greater AR_s, indicating the presence of several elongated units within the layout. The allocation uniformity metrics AU_R, AU_S, and AU_U measure the balance of space allocation across different hierarchical levels. Both the *Non-BA* and *RCVT* methods demonstrate superior performance at all levels, indicating more uniform and balanced space division. Furthermore, the ablation results reveal that our proposed method significantly enhances *Space Utilization*, achieving better efficiency in space usage. This improvement highlights the effectiveness of our approach in optimizing layout design and ensuring efficient use of available space.

6.5. Application

Design of atrium. Atriums are common architectural features in commercial space layouts, providing spaces for exhibitions, social gatherings, and entertainment activities. In this method, atriums are considered strict constraints that influence the overall layout design. By strategically placing atriums within the layout, different types of sites can be synthesized around them, resulting in sub-regions. To ensure flexibility and customization, users can specify the location, number, and shape of atriums within the layout. This empowers them to add atriums of fixed shapes and sizes that align with their preferences and design goals. The layout generation process considers the specified atriums and generates a layout that seamlessly integrates them with the surrounding units, as shown in Fig. 15.

Extreme test. In real-world situations, the conditions tend to be more complex, and the boundaries may present a wide variety of cases. Fig. 16 showcases the results generated by our method in some special cases. These extreme scenarios highlight the adaptability of the layout generation process in handling complex and irregular boundary shapes.

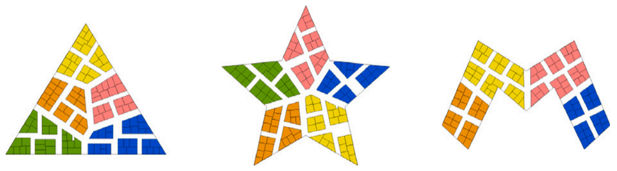


Fig. 16. Extreme test. Here show the commercial space layouts generated with special boundary conditions, which are used to test the performance and robustness of our method.

Diversity of generation. The position of sites plays a pivotal role in determining the distribution of elements within the layout, consequently yielding varied layout outcomes. If we allow for broader threshold values and generate various sites, the division of regions will also vary accordingly. With the exact input and constraints, our method can generate multiple distinct outcomes, offering users a range of choices. Fig. 17 illustrates the diverse results produced by our method. This demonstrates the method's ability to produce diverse and varied layouts while satisfying the given design requirements.

Synthetic dataset. We extracted boundaries from the RPLAN dataset [7]. These boundaries are represented by polygons, from which we selectively removed certain points to create boundaries that include slanted edges. This process was used to simulate the boundaries of a commercial area, which then served as the input for our dataset generation. Using our method, we created a synthetic dataset containing approximately 3000 vector-based layouts, as shown in Fig. 18. The synthetic dataset plays a crucial role in our study by providing a diverse range of layout configurations that help in testing and refining our algorithms. Additionally, the dataset aids in training machine learning models to recognize and optimize various layout patterns, contributing to the field of architectural design.

7. Conclusion

We propose a method for automatically generating commercial space layouts hierarchically. Real commercial areas often face complex situations, and our method can provide a novel generation framework for the conceptual design of commercial space layouts that architects can refine into a practical layout. In our approach, we use various Voronoi diagrams to divide the areas and utilize the capabilities of centroidal optimization to optimize the distribution of the regions, resulting in more appropriate layouts. By incorporating constraints on the regions, we can generate layouts that meet specific design requirements. To evaluate the quality of the generated layouts, we employ various parameter metrics, which allow us to quantitatively assess the layouts based on factors such as visual appeal and crowd evacuation. The hierarchical approach for generating layouts provides architects and designers with a powerful tool to assist them in creating well-designed commercial space layouts by effectively exploring and applying various constraint options.

Our work could be improved for future exploration. There is still potential for improvement in the speed of our layout generation process. We can explore various approaches, such as utilizing concurrent computation with different regions, to further enhance the speed and efficiency of our method. Moreover, we can investigate the applicability of our method in diverse domains, including urban layout planning and park design. By adapting and extending our approach, we can address specific challenges and requirements unique to these contexts, contributing to more effective and optimized layouts. Additionally, our method can potentially serve as a tool for generating datasets. By utilizing the layouts generated through our approach, we can explore the possibilities of employing deep learning techniques for layout design.

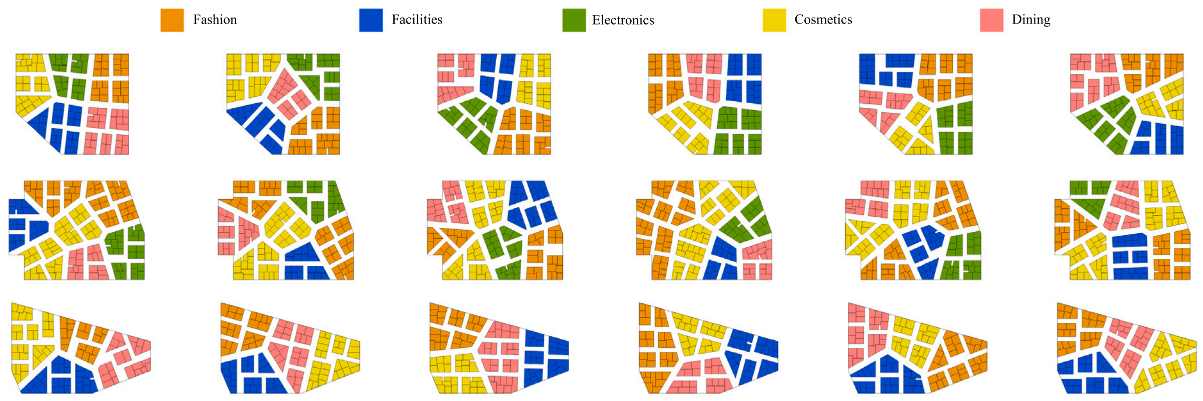


Fig. 17. Diversity of generation. Here shows multiple commercial space layouts generated by our method under the same boundary and constraints.

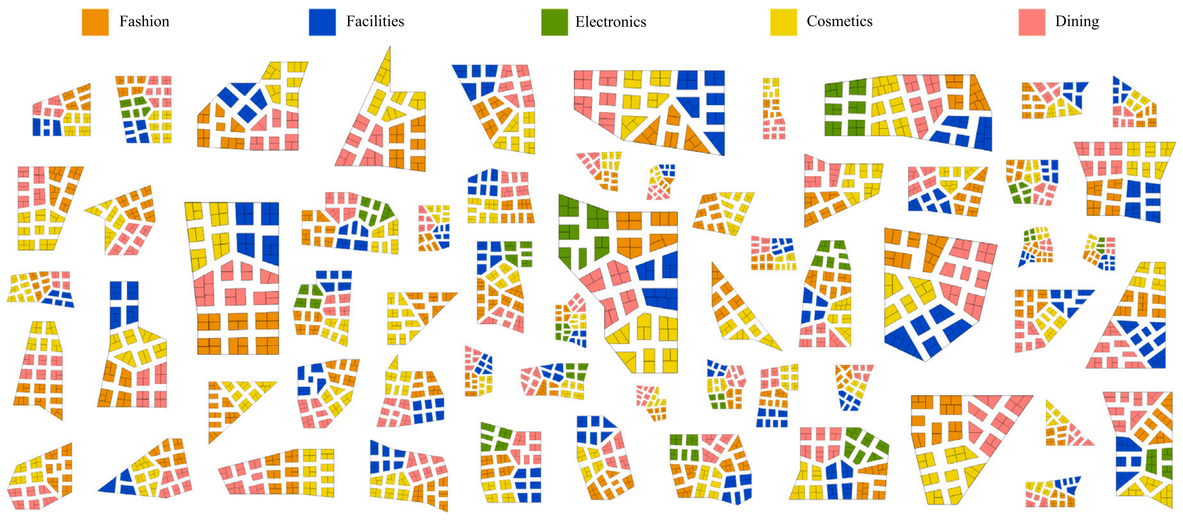


Fig. 18. Synthetic dataset. Here showcases a portion of the synthetic dataset generated using our method.

CRediT authorship contribution statement

Yuntao Wang: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology. **Wenming Wu:** Writing – review & editing, Writing – original draft, Methodology, Funding acquisition, Conceptualization. **Yue Fei:** Writing – review & editing, Writing – original draft, Validation. **Liping Zheng:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62102126, 62372152), the Fundamental Research Funds for the Central Universities of China (JZ2023HGTB0269) and the National Key Research and Development Program of China (2022YFC 3900800).

Appendix A. Hierarchical spatial division

We divide the layout into different levels and perform region division within each level. In the experiment, we use a three-level hierarchy: *Region-Subregion-Unit*, as shown in Fig. A.19. The region division is closely tied to the sites, allowing users to assign different levels to the sites to determine their grouping. For instance, at the *Region* level, areas can be divided into various regions based on type, such as fashion, electronics, and dining. At the *Subregion* level, further subdivision can categorize these regions into specific types, such as home appliances or digital products. At the *Unit* level, we can break them down into individual shops, ensuring that related units are adjacent and maintain their relevance.

Region generation. *Region* represents a large area within the layout. During the *Region* generation stage, the CVT is used for division, and paths are added. The distances from the edges of two adjacent cells to their corresponding sites are equal in the Voronoi diagram. Therefore, we consider the edges adjacent to *Regions* as accessible paths, providing suitable paths for the layout. We set a specific width for the paths based on the current level and contract the surrounding *Regions* to form actual paths. The intersections of paths at this level are highly likely to be focal points where pedestrian traffic converges. These areas present ideal opportunities for installing automatic escalators to enhance mobility and optimize the flow of individuals.

Subregion generation. *Subregion* represents the secondary subdivision area within the commercial space, indicating the specific subcategory of goods in a particular area. During this stage, we choose the RCVT for optimization and region division. The adjacent edges between *Subregion* are added to the layout as paths. In some cases, such as in a narrow region, a *Subregion* may block the connectivity of the paths. To ensure the connectivity of the paths, we treat the edges of this *Subregion*

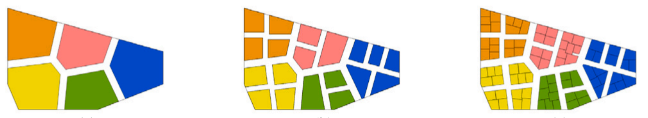


Fig. A.19. Hierarchical spatial division. (a) *Region*. (b) *Subregion*. (c) *Unit*. Different widths can be set to paths according to the level.

adjacent to the boundary as accessible paths. We consider the gaps formed by the calculation to be accessible areas, and gaps that may occur between *Subregions* at this level can serve as shared areas at path intersections.

Unit generation. *Unit* represents the smallest unit within the commercial space. During the *Unit* generation stage, the RCVT is selected for division. As it is the final level, adding region boundary paths is unnecessary. After the region generation is complete, adjustments are made to the *Unit* polygons. We divide the gaps into smaller shapes based on their boundaries and merge smaller shapes into the surrounding regions. The *Units* along the overall layout boundary or internal paths are considered accessible. For *Units* within a *Subregion* that are not accessible, we use the shortest path algorithm to find its connection to the accessible paths along the edges of *Units* and construct an inward concave path to make the *Units* accessible.

Appendix B. Layout optimization

After dividing a region using Rectangular Centroidal Voronoi Tessellation (RCVT), we obtain a set of sub-regions $R = \{r_i\}_{i=1}^{n}$ and their corresponding sites $S = \{s_i\}_{i=1}^{n}$. However, this initial division may result in polygonal gaps $G = \{g_i\}_{i=1}^{m}$ between the sub-regions, and some sub-regions may become inaccessible due to their configuration. To address these issues, we input the sites into a simulated annealing optimization algorithm. This algorithm iteratively adjusts the positions of the sites, aiming to achieve a spatial division that satisfies specific objectives, such as minimizing gaps and ensuring the accessibility of all sub-regions.

Space utilization optimization. In certain scenarios, achieving higher space utilization is a priority. To this end, we aim to minimize the area of the gaps between the sub-regions. The objective function is defined as the total area of the gaps:

$$E_{\text{uti}} = \sum_{i=1}^{m} \text{Area}(g_i) \tag{B.1}$$

Here g_i is the polygonal gap among sub-regions. $Area(\cdot)$ is the operator for area calculation.

Accessibility optimization. Accessibility refers to whether each sub-region can be accessed via a connected path. Given a set of accessible paths $P = \{p_i\}$ and a set of sub-regions $S = \{s_i\}$, a sub-region is considered accessible if it is adjacent to any accessible path. The objective function for accessibility is defined as follows:

$$E_{\text{acc}} = \mathcal{K} N_{\text{inacc}} + \sum_{i} \|s_i - p_i\| \tag{B.2}$$

Here N_{inacc} represents the number of inaccessible sub-regions, and $\mathcal{K}$ is a large constant used to penalize inaccessibility. s_i refers to the site corresponding to an inaccessible sub-region, while p_i denotes the nearest accessible path to s_i. The term $\|s_i - p_i\|$ represents the distance from the site s_i to its nearest accessible path. The optimization continues until all sub-regions are adjacent to accessible paths.

Mixed optimization. By combining the objectives of maximizing space utilization and accessibility, we achieve a more comprehensive and desirable layout refinement. The mixed optimization considers both the minimization of gaps (to enhance space utilization) and the accessibility of all sub-regions. The combined objective function is formulated as follows:

$$E = \omega_1 E_{\text{uti}} + \omega_2 E_{\text{acc}} \tag{B.3}$$

Here, ω_1 and ω_2 are weights that balance the influence of the space utilization and accessibility objectives, respectively.

Appendix C. Site-constrained division

Location constraint. Altering the positions of sites causes the corresponding regions to adjust accordingly. By adding, removing, or relocating sites, we can dynamically control the layout of the regions. This flexibility allows for fine-tuning of the division process, enabling the layout to better accommodate specific design requirements and spatial constraints.

Size constraint. The number of regions at each level is determined by the user-specified number of sites, ensuring that the resulting regions are consistent with the size, distribution, and arrangement of the sites. Additionally, by imposing quantity constraints at various levels, we can rigorously ensure that the total number of Units matches the specified target, providing precise control over the final layout and enabling greater flexibility in design customization.

Function constraint. Users can specify the number of different functions and apply function constraints to specific regions. This includes the ability to assign distinct functions to individual regions or set broader functional requirements for entire sections of the layout. By controlling the arrangement of functions, users can tailor the spatial organization to meet specific design goals, ensuring that functional areas are appropriately distributed and aligned with the intended use of the space.

Capacity constraint. The capacity of a space is often directly related to its area. During the computation process, the area of a region can be adjusted by manipulating the weight of its corresponding site. Increasing the weight of a site results in a larger capacity for its corresponding region. In commercial space layouts, each unit may have distinct attributes, such as function, size, or usage priority, which can influence the weight assigned to its site. Users can set site weights based on factors like estimated profitability, operational importance, or required floor space. This approach provides precise control over the layout, allowing for customized spatial allocation that aligns with design objectives and business goals.

Data availability

Data will be made available on request.

References

[1] He L, Aliaga D. GlobalMapper: Arbitrary-shaped urban layout generation. In: Proceedings of the IEEE/CVF international conference on computer vision. 2023, p. 454–64.
[2] Sun J, Wu W, Liu L, Min W, Zhang G, Zheng L. Wallplan: synthesizing floorplans by learning to generate wall graphs. ACM Trans Graph 2022;41(4):1–14.
[3] Kong X, Jiang L, Chang H, Zhang H, Hao Y, Gong H, et al. Blt: Bidirectional layout transformer for controllable layout generation. In: European conference on computer vision. Springer; 2022, p. 474–90.
[4] Para WR, Guerrero P, Mitra N, Wonka P. COFS: Controllable furniture layout synthesis. In: ACM SIGGRAPH 2023 conference proceedings. 2023, p. 1–11.
[5] Sun J, Zheng L, Zhang G, Wu W. BubbleFormer: Bubble diagram generation via dual transformer models. Comput Graph Forum 2023;42:e14984.

[6] Feng T, Yu L-F, Yeung S-K, Yin K, Zhou K. Crowd-driven mid-scale layout design. ACM Trans Graph 2016;35(4):1–14, 132.

[7] Wu W, Fu X-M, Tang R, Wang Y, Qi Y-H, Liu L. Data-driven interior plan generation for residential buildings. ACM Trans Graph 2019;38(6):1–12.

[8] Wang S, Zeng W, Chen X, Ye Y, Qiao Y, Fu C-W. ActFloor-GAN: Activity-guided adversarial networks for human-centric floorplan design. IEEE Trans Vis Comput Graphics 2021;29(3):1610–24.

[9] Upadhyay A, Dubey A, Mani Kuriakose S, Agarawal S. Floorgan: Generative network for automated floor layout generation. In: Proceedings of the 6th joint international conference on data science & management of data (10th ACM IKDD CODS and 28th COMAD). 2023, p. 140–8.

[10] Hua H, Hovestadt L, Tang P, Li B. Integer programming for urban design. European J Oper Res 2019;274(3):1125–37.

[11] Qin Y, Zhao N, Sheng B, Lau RW. Text2City: One-stage text-driven urban layout regeneration. In: Proceedings of the AAAI conference on artificial intelligence. Vol. 38, (5):2024, p. 4578–86.

[12] Peng C-H, Yang Y-L, Wonka P. Computing layouts with deformable templates. ACM Trans Graph 2014;33(4):1–11.

[13] Wu W, Fan L, Liu L, Wonka P. MIQP-based layout design for building interiors. Comput Graph Forum 2018;37(2):511–21.

[14] Yuan Y, Liu G, Dang R, Lau SSY, Qu G. Architectural design and consumer experience: an investigation of shopping malls throughout the design process. Asia Pac J Mark Logist 2021;33(9):1934–51.

[15] Liu H, Yang Y-L, AlHalawani S, Mitra NJ. Constraint-aware interior layout exploration for pre-cast concrete-based buildings. Vis Comput 2013;29:663–73.

[16] Fisher M, Savva M, Li Y, Hanrahan P, Nießner M. Activity-centric scene synthesis for functional 3D scene modeling. ACM Trans Graph 2015;34(6):1–13.

[17] Laignel G, Pozin N, Geffrier X, Delevaux L, Brun F, Dolla B. Floor plan generation through a mixed constraint programming-genetic optimization approach. Autom Constr 2021;123:103491.

[18] Yu L-F, Yeung S-K, Terzopoulos D. The clutterpalette: An interactive tool for detailing indoor scenes. IEEE Trans Vis Comput Graphics 2015;22(2):1138–48.

[19] Zhang S-K, Tam H, Li Y, Ren K-X, Fu H, Zhang S-H. Scenedirector: Interactive scene synthesis by simultaneously editing multiple objects in real-time. IEEE Trans Vis Comput Graphics 2023.

[20] Zhang S-K, Huang J, Yue L, Zhang J-T, Liu J-H, Lai Y-K, et al. SceneExpander: Real-time scene synthesis for interactive floor plan editing. In: Proceedings of the 32nd ACM international conference on multimedia. 2024, p. 6232–40.

[21] Zhang S-K, Li Y-X, He Y, Yang Y-L, Zhang S-H. Mageadd: Real-time interaction simulation for scene synthesis. In: Proceedings of the 29th ACM international conference on multimedia. 2021, p. 965–73.

[22] Sun J, Yu X, Baciu G, Green M. Template-based generation of road networks for virtual city modeling. In: Proceedings of the ACM symposium on virtual reality software and technology. 2002, p. 33–40.

[23] Galin E, Peytavie A, Maréchal N, Guérin E. Procedural generation of roads. Comput Graph Forum 2010;29(2):429–38.

[24] Aliaga DG, Vanegas CA, Benes B. Interactive example-based urban layout synthesis. ACM Trans Graph 2008;27(5):1–10.

[25] Nishida G, Garcia-Dorado I, Aliaga DG. Example-driven procedural urban roads. Comput Graph Forum 2016;35.

[26] Yang Y-L, Wang J, Vouga E, Wonka P. Urban pattern: Layout design by hierarchical domain splitting. ACM Trans Graph 2013;32(6):1–12.

[27] Vanegas CA, Aliaga DG, Benes B, Waddell PA. Interactive design of urban spaces using geometrical and behavioral modeling. ACM Trans Graph 2009;28(5):1–10.

[28] Das S, Day C, Hauck A, Haymaker J, Davis D. Space plan generator. In: Acadia conference. 2016, p. 106–15.

[29] aha N, Haymaker J, Shelden D. Space allocation techniques(SAT). In: Acadia conference. 2020, p. 248–57.

[30] Chu H, Li D, Acuna D, Kar A, Shugrina M, Wei X, et al. Neural turtle graphics for modeling city road layouts. In: Proceedings of the IEEE/CVF international conference on computer vision. 2019, p. 4522–30.

[31] Vanegas CA, Garcia-Dorado I, Aliaga DG, Benes B, Waddell P. Inverse design of urban procedural models. ACM Trans Graph 2012;31(6):1–11.

[32] Vanegas CA, Kelly T, Weber B, Halatsch J, Aliaga DG, Müller P. Procedural generation of parcels in urban modeling. Comput Graph Forum 2012;31(2pt3):681–90.

[33] Chen Z, Song P, Ortner FP. Hierarchical co-generation of parcels and streets in urban modeling. Comput Graph Forum 2024;43(2):e15053.

[34] Fahmy SA, Alablani BA, Abdelmaguid TF. Shopping center design using a facility layout assignment approach. In: 2014 9th international conference on informatics and systems. 2014, ORDS–1–ORDS–7.

[35] Zhang S-K, Liu J-H, Li Y, Xiong T, Ren K-X, Fu H, et al. Automatic generation of commercial scenes. In: Proceedings of the 31st ACM international conference on multimedia. 2023, p. 1137–47.

[36] Zhang B-z, Li B. Exploration on the application of geometric prototype and form encoding strategies in the layout generation of commercial buildings. In: CDAC. 2022, p. 105–14.

[37] Nowak A. Application of voronoi diagrams in contemporary architecture and town planning. Challenges Mod Technol 2015;6(2):30–4.

[38] Guo P, Zhang F, Wang H, Qin F. Suitability evaluation and layout optimization of the spatial distribution of Rural Residential Areas. Sustainability 2020;12(6).

[39] Al-Dahhan MRH, Schmidt KW. Voronoi boundary visibility for efficient path planning. IEEE Access 2020;8:134764–81.

[40] Chi W, Ding Z, Wang J, Chen G, Sun L. A generalized voronoi diagram-based efficient heuristic path planning method for RRTs in mobile robots. IEEE Trans Ind Electron 2022;69(5):4926–37.

[41] Huang S-K, Wang W-J, Sun C-H. A path planning strategy for multi-robot moving with path-priority order based on a generalized voronoi diagram. Appl Sci 2021;11(20).

[42] Tahilyani S, Saxena S, Karras DA, Kant Gupta S, Kumar Dixit C, Haralayya B. Deployment of autonomous vehicles in agricultural and using voronoi partitioning. In: 2022 international conference on knowledge engineering and communication systems. ICKES, 2022, p. 1–5.

[43] Wang F, Ye M, Zhu H, Gu D. Optimization method for conventional bus stop placement and the bus line network based on the voronoi diagram. Sustainability 2022;14(13).

[44] Yao Y, Wu W, Zhang G, Xu B, Zheng L. Power diagram based algorithm for the facility location and capacity acquisition problem with dense demand. Front Comput Sci 2022;16(6):166709.

[45] Lei H-Y, Li J-R, Xu Z-J, Wang Q-H. Parametric design of voronoi-based lattice porous structures. Mater Des 2020;191:108607.

[46] Vavilova I, Elyiv A, Dobrycheva D, Melnyk O. The Voronoi tessellation method in astronomy. Intell Astrophys 2021;57–79.

[47] Ma N, Zhang H, Hu H, Qin Y. ESCVAD: An energy-saving routing protocol based on voronoi adaptive clustering for wireless sensor networks. IEEE Internet Things J 2022;9(11):9071–85.

[48] Richter A, Ng KTW, Karimi N. Meshing Centroidal Voronoi Tessellation with spatial statistics to optimize waste management regions. J Clean Prod 2021;295:126465.

[49] Larrea M, Urribarri D, Martig S, Castro S. Spherical layout implementation using centroidal voronoi tessellations. 2009, arXiv preprint arXiv:0912.3974.

[50] Abedi M, Wichman R. Cellular network planning under variable QoS requirements using voronoi algorithm. In: 2020 IEEE wireless communications and networking conference. WCNC, 2020, p. 1–6.

[51] Abedi M, Wichman R. Planning and optimization of cellular networks using load-based voronoi algorithm. In: 2018 52nd asilomar conference on signals, systems, and computers. 2018, p. 1589–93.

[52] Yadav N, Kang JH, Rybkowski ZK, Yarnold MT, He W. Stress-weighted centroidal voronoi tessellation for structural design. Finite Elem Anal Des 2023;216:103905.

[53] D'Acunto M, Moroni D, Puntoni A, Salvetti O. Optimized dislocation of mobile sensor networks on large marine environments using voronoi partitions. J Mar Sci Eng 2020;8(2).

[54] Wu H-Y, Nöllenburg M, Viola I. Multi-level area balancing of clustered graphs. IEEE Trans Vis Comput Graphics 2022;28(7):2682–96. http://dx.doi.org/10.1109/TVCG.2020.3038154.

[55] Chen Z-S, Hou K-R, Zhu M-Y, Xu Y, Zhu Q-X. A virtual sample generation approach based on a modified conditional GAN and centroidal voronoi tessellation sampling to cope with small sample size problems: Application to soft sensing for chemical process. Appl Soft Comput 2021;101:107070.

[56] Wang L, Hétroy-Wheeler F, Boyer E. A hierarchical approach for regular centroidal voronoi tessellations. Comput Graph Forum 2016;35.

[57] Xin S-Q, Lévy B, Chen Z, Chu L, Yu Y, Tu C, et al. Centroidal power diagrams with capacity constraints: Computation, applications, and extension. ACM Trans Graph 2016;35(6):1–12.

[58] Choi S-G, Kyung C-M. A floorplanning algorithm using rectangular voronoi diagram and force-directed block shaping. In: 1991 IEEE international conference on computer-aided design digest of technical papers. IEEE Computer Society; 1991, p. 56–7.

[59] Chatzikonstantinou I. A 3-dimensional architectural layout generation procedure for optimization applications: DC-RVD. In: Proceedings of the 2014 eCAADe conference. Vol. 1, 2014, p. 287–96.

[60] Wang Y-C, Liu J, Lin F, Seah H-S. Generating orthogonal voronoi treemap for visualization of hierarchical data. In: Computer graphics international conference. Springer; 2020, p. 394–402.

[61] Yu-you Y, Gao-feng Z, Ben-zhu X, Li-ping Z. Computation method of variable capacity constrained centroidal power diagram. J Graph 2021;42(3):492.

[62] Lloyd S. Least squares quantization in PCM. IEEE Trans Inform Theory 1982;28(2):129–37.

[63] The CGAL Project. CGAL user and reference manual. 5.6. CGAL Editorial Board; 2023, URL https://doc.cgal.org/5.6/Manual/packages.html.

[64] depthmapX development team. depthmapX. 2020, URL https://github.com/SpaceGroupUCL/depthmapX/.

[65] Batty M. A new theory of space syntax. Centre for Advanced Spatial Analysis (UCL); 2004.

[66] Fezzai S, Fares RB, Boutouata FE, Benachi N. Investigating the impact of spatial configuration on users' behaviour in shopping malls case of bab-ezzouar shopping mall in algiers. Int J Built Environ Sustain 2020;7(3):23–35.
[67] Andi A, Abednego IA, Gultom BJB. A space syntax guide to optimize shopping mall: A systematic review. Int J Environ Archit Soc 2021;1(01):19–30.
[68] Hillier B, Hanson J. The social logic of space. Cambridge University Press; 1989.
[69] Hillier B. Space is the machine: a configurational theory of architecture. Space Syntax; 2007.
[70] Van Nes A, Yamu C. Introduction to space syntax in urban studies. Springer Nature; 2021.