Multimed Tools Appl (2019) 78:5051–5079
https://doi.org/10.1007/s11042-018-6334-5

Web3D-based automatic furniture layout system using recursive case-based reasoning and floor field

Peihua Song[1] · **Youyi Zheng**[2] · **Jinyuan Jia**[1] · **Yan Gao**[3]

Received: 1 October 2017 / Revised: 3 June 2018 / Accepted: 26 June 2018 /
Published online: 5 July 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract Furniture layout in a virtual 3D scene is an important and challenging task, as it is time-consuming and requires experience. To address this issue, we propose automatic furniture layout algorithms to help users to rapidly generate reasonable layout results. Specifically, our algorithms divide a scene layout into four layout modes, namely, coupled mode, enclosed mode, matrix mode, and circular mode. Then each model is solved independently. The coupled mode is solved using recursive techniques and case-based reasoning. The enclosed mode is solved using floor field. The distance and angle among the furniture are determined by ergonomics guidelines. Finally, the layout results of the scene can be obtained by combining the solutions from these layout modes, and an evaluation method for the layout results based on user feedback is proposed. For a room with non-rectangular floor, our algorithms can also handle this case using shape standardization techniques. Based on our algorithms, an online 3D furniture layout system is developed. Many experiments are conducted on the system with the real interior design cases, and we compared our algorithms with other popular algorithms. The experimental results show that our algorithms are efficient and can meet the real response requirements of online furniture layout.

Keywords Furniture layout · Case-based reasoning · Floor field · Recursive techniques

✉ Jinyuan Jia
jiajytj@126.com

Peihua Song
sph2000@126.com

Youyi Zheng
youyizheng@zju.edu.cn

Yan Gao
gaoy@clarkson.edu

[1] School of Software Engineering, Tongji University, Shanghai 201804, People's Republic of China

[2] State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310058, People's Republic of China

[3] Electrical and Computer Engineering, Clarkson University, Potsdam, NY 13676, USA

1 Introduction

The scene design can be widely found in many domains such as city planning, architectural design and 3D scene modeling [6, 18, 20, 28]. It is also an active research area in computer graphics and optimization techniques. As an important component of scene design, furniture layout has gained more attention in recent years, often existing in 3D interior design and game scene modeling. Traditional manual furniture layout is an uneasy task because it is time-consuming and tedious. Furniture layout is often performed in a virtual 3D environment and each piece of furniture has six degrees of freedom. Thus it is inconvenient to manually manipulate the furniture with a mouse. Moreover, for users without interior design expertise and experience, they need many efforts to get a reasonable furniture layout result; for interior designers, repeating the similar operation of arranging furniture is also a tedious work.

To overcome these problems, researchers have proposed several efficient algorithms for automatic furniture layout, including algorithms based on optimization model [1, 5, 17, 19, 23, 24, 32], constraints [4, 9, 13, 26, 29] and case library [2, 7, 27, 30, 31]. However, most of existing algorithms of furniture layout have a limitation either in the number of furniture or in the shape of the layout room. For example, Yu et al. [32] defined a layout evaluation function, and then searched the suitable location to arrange furniture using simulated annealing. But for the layout problem of the large quantity of furniture and area of the room, the number of iterations increase and the convergence speed becomes slow as the large search range, resulting in a long run time [17, 32]. Algorithms based on case library or constraints rarely consider the impact of the doors and windows on the layouts [2, 13, 27], requiring the floor shape of the room to be rectangular. Meanwhile, with the rapid development of 3D technique in web environment [10, 22, 33], there emerge some online 3D interior design systems. These systems are popular because users can perform an online design without installing any offline software. This trend motivates us to extend existing offline furniture layout to online furniture layout, and online furniture layout requires that the furniture layout algorithms have real time response and can generate reasonable layout results.

In this paper, we investigate the problem of automatic furniture layout in the Web3D-based environment. The general framework of our algorithms is as follows. The whole room layout is divided into four furniture layout modes, namely, coupled mode, enclosed mode, matrix mode, and circular mode, and then these modes are solved separately. The furniture layout mode represents the layout relationship between the furniture and furniture or a room, and the four layout modes will be described in details in Section 3. Then we combine the solutions from these modes to obtain the final layout result of the room. The major contributions of this paper can be summarized as follows.

(1) We propose the respective shape standardization algorithms for the bedroom and living room, which is the preprocessing for a non-rectangular room.
(2) To reduce layout complexity, we introduce the four furniture layout modes mentioned previously and present the respective algorithms for them. All these algorithms utilize ergonomics guidelines to determine the distance and angle among the furniture. Coupled mode and enclosed mode are solved using recursive case-based reasoning and floor field, respectively.
(3) Our framework is highly extensive that we have added four furniture layout algorithms for the living room, the bedroom, the classroom, and the restaurant.
(4) A Web3D-based interactive system for automatic furniture layout is presented based on our algorithms. The system can yield reasonable results of furniture layout when users press the specified button in the system. In addition, we also present an evaluation method for the

layout results based on user feedback. Our algorithms have been implemented with real interior design cases, and compared with other algorithms. The experimental results show that the algorithms are effective, and can meet the real response requirements of online layout.

The rest of this paper is organized as follows. Section 2 reviews related work. In Section 3, we describe the four layout modes in detail and give a systematic overview. Section 4 presents the shape standardization algorithms for the bedroom and the living room. Section 5, 6, and 7 describe respective algorithms for the four layout modes. Section 8 presents the respective furniture algorithms for various rooms based on the general framework. Section 9 introduces a valuation method for layout results based on user feedback. Section 10 reports the experimental results and gives a discussion, and finally, we conclude our paper in Section 11.

2 Related work

The virtual 3D scene design and synthesis have been widely used in the outdoor scene. For example, Merrell et al. [18] proposed an approach for automatically generation of building layouts using a Bayesian network. Esch et al. [6] presented a framework for modeling large street networks. The user can create street network from scratch using the framework. In this section, we focus on the related work of the furniture layout methods in the indoor scene,and these methods can be broadly divided into three categories as follows.

2.1 Algorithms based on optimization model

Many intelligent optimization algorithms are used to solve the furniture layout problem, such as, simulated annealing algorithm, genetic algorithm and particle swarm optimization algorithm. The general framework of these algorithms is as follows. First these algorithms define an evaluation function by furniture layout rules which include the distance and angle among the furniture. Then, the layout results are obtained by searching the solutions using intelligent algorithms. Yu et al. [32] presented an algorithms based on the simulated annealing algorithm. The evaluation function of the algorithm mainly considers accessibility, visibility, pathway constraints and so on. Merrell et al. [19] presented an interactive furniture layout system based on interior design guidelines. The algorithm incorporates the layout guidelines as terms in a density function, and then generates layout suggestions. Chen et al. [5] presented a hierarchical optimization strategy, and the strategy combined with particle swarm to solve the furniture layout problem, which improves computing efficiency. Akase et al. [1] introduced a furniture layout cost function based on ergonomic, and used interactive evolutionary to solve the cost function. Liu et al. [17] introduced a composite model for home furnishing generation and used the genetic algorithm to solve the model.

Although these algorithms can generate reasonable results, the running time of them will increase rapidly as a scene contains a lot of furniture. To reduce layout complexity and running time, in this paper we divide a sense layout into fours layout modes and solve the layout modes respectively.

2.2 Algorithms based on constraints

Bukowski et al. [4] presented a software framework to help designers manipulate 3D objects with a 2D cursor. The framework used the object associations to determine the location of each object. Smith et al. [26] also presented techniques to restrict object motion to assist users in the

3D environment, which considered various constraints among the objects, such as dual constraints. Although [4, 26] considered some geometric and physical constraints in designing, their whole design process was user-driven and still time-consuming. Kjølaas [13] used a set of default layout templates to solve the furniture layout problem. The algorithm is limited to the rectangular room. Germer et al. [9] presented an approach to arrange furniture in large virtual indoor scenes using the agent-based interiors, but the furniture layout case of multiple doors in a room was not considered in this approach.

2.3 Algorithms based on case library

Fisher et al. [7] presented a method for synthesizing 3D objects arrangements from examples. The method synthesizes a diverse set of plausible new scenes using a machine learning method. Xu et al. [30] proposed a framework that automatically created a 3D scene from a sketch drawing. The framework obtains the corresponding 3D model by searching the model library. Xu et al. [31] presented a system for automatically synthesizing 3D interior scenes for a room shape entered by users. The algorithm arranges 3D models based on the relationships between the models and room. In the authors' previous work [27], an algorithm for furniture layout was proposed, but the algorithm is also limited to the rectangular shape room.

These algorithms rarely consider the layout problem of the non-rectangular room with multiple doors. Indeed, there exist many living rooms which are non-rectangular rooms and have multiple doors. To deal with these problems, this paper adopts shape standardization techniques and case-based reasoning.

3 Furniture layout modes and system overview

Furniture layout is a complicated problem. On the one hand, there are many constraints among furniture in layout scenes. On the other hand, layout scenes have different functions and floor shapes. If the problem is solved directly, the process of solving the problem is quite complex or takes a long time. To reduce the complexity of the problem, we first introduce four furniture layout modes using the idea of divide and conquer in this section. Based on the four furniture layout modes, we also give the overview of the proposed system.

3.1 Furniture layout modes

A furniture layout mode refers to the layout relationship between the furniture and furniture or the room. We define four layout modes, namely, coupled mode, enclosed mode, matrix mode and circular mode. (1) Coupled mode requires that a certain distance and angle among furniture need to be maintained between the furniture, such as the bed and bedside cabinets (see Fig. 1a). It can be divided into two categories: furniture and furniture coupled mode, furniture and room coupled mode. (2) Enclosed mode requires that the furniture first is arranged at the corner, followed by along the wall, such as a writing table in the room (see Fig. 1b). (3) Matrix mode requires the furniture to be arranged in a rectangular region, such as the desks in the classroom (see Fig. 1c). (4) Circular mode requires the furniture to be arranged around circular furniture, such as the circular dining table and chairs (see Fig. 1d). The distance and angle between furniture in these layout models need to meet the ergonomic [16]. Common furniture layout modes presented in indoor scenes are show in Table 1.

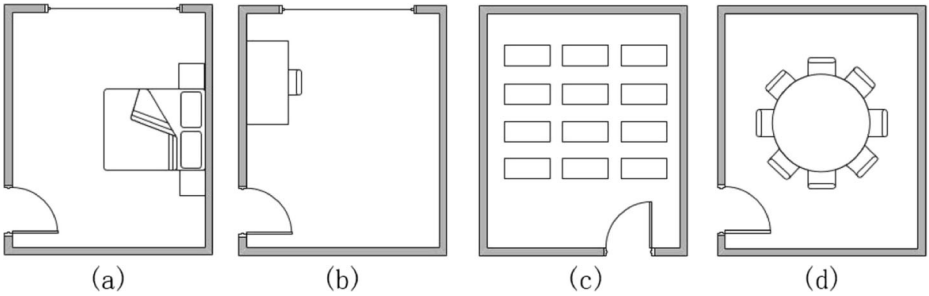


Fig. 1 **a** An example of the coupled mode **b** An example of the enclosed mode **c** An example of the matrix mode **d** An example of the circular mode

3.2 System overview

The main function of the system is to automatically generate a furniture layout result to help user complete the work of interior design. The system is developed using the WebGL and C# language in Visual Studio 2015. As shown in Fig. 2, the system consists of three modules: user input module, automatic furniture layout module and results output module.

The functions of the three modules are listed below. (1) The function of user input module is as follows. A user can input a 2D floor plan, and then the system generates a 3D room and interactive environment. Next, the user can add, move, delete, and rotate the 3D furniture in the room. (2) Automatic furniture layout module is the core module of the system, which implements the respective furniture layout algorithms for living room, bedroom, classroom, and restaurant. (3) In the results output module, the user can adjust the camera position and angle, obtaining a 2D picture of layout results from the current 3D environment.

We have given a simplified description for the general framework of our algorithms in Section 1, and here we give a detailed description of the framework. The framework is a general process of furniture layout. Thus we need to adjust these phases according to the function of the scene. The framework is composed of 4 steps as follows.

Step 1: If the floor shape of the room is non-rectangular, then we extract the rectangular layout region from the floor shape to reduce the complexity of the whole room layout.

Step 2: According to the type of the room inputted by the user, the room layout is split into four furniture layout modes, namely, coupled mode, enclosed mode, matrix mode, and circular mode.

Step 3: Solve these selected modes respectively. The coupled mode is solved using recursive techniques and case-based reasoning. The enclosed mode is solved using floor field. The distance and angle among furniture can be set according to ergonomics.

Table 1 Common layout modes presented in indoor scenes

Scene	Layout modes
Living room	Coupled mode and enclosed mode
Bedroom	Coupled mode and enclosed mode
Classroom	Coupled mode and matrix mode
Restaurant	Matrix mode, circular mode and coupled mode

Step 4: Merge the solutions from these layout modes into a solution, namely, the layout result of the room.

4 Room shape standardization

In the process of furniture layout, it is common that the floor shape of the bedroom and the living room are non-rectangular. These rooms contain aisles, windows regions, and the regions for arranging furniture. Indeed, furniture isn't arranged in the regions of aisles and windows generally. To reduce the search regions and computation time of the layout algorithms, we extract a rectangular region from the layout room that does not contain aisles and windows regions, and this process is called room shape standardization.

4.1 Region split points

The function of region split points is to construct rectangular regions. To illustrate how to generate a region split point, first we give the following notions. Let the polygon $P = p_1p_2 \ldots p_n$ denote the floor shape of the bedroom or living room (see Fig. 3a). The polygon P contains two sets: a set of points $\{p_1, p_2 \ldots p_n\}$ and a set of edges $\{p_1p_2, p_2p_3 \ldots p_{n-1}p_n, p_np_1\}$. The edges denote the walls in the room. The point p_{door} and the point p_{window} denote the position of the door and the window, respectively. The approach to generate a region split point is as follows.

Step 1: Add the X coordinate and the Z coordinate of each point of the set $\{p_1, p_2 \ldots p_n\}$ to an array A and an array B, respectively.
Step 2: Remove the same elements in A and B, respectively. Sort A and sort B in increasing order.
Step 3: Randomly Pick an index i and an index j from A and B, respectively. Generate a point $(A[i], B[j])$ and return it.

We define the point $(A[i], B[j])$ as a region split point. The region split points can be divided into two groups: internal region split points and external region split points. The former group requires that the split points are on the edges of P or inside the polygon P, and the latter group

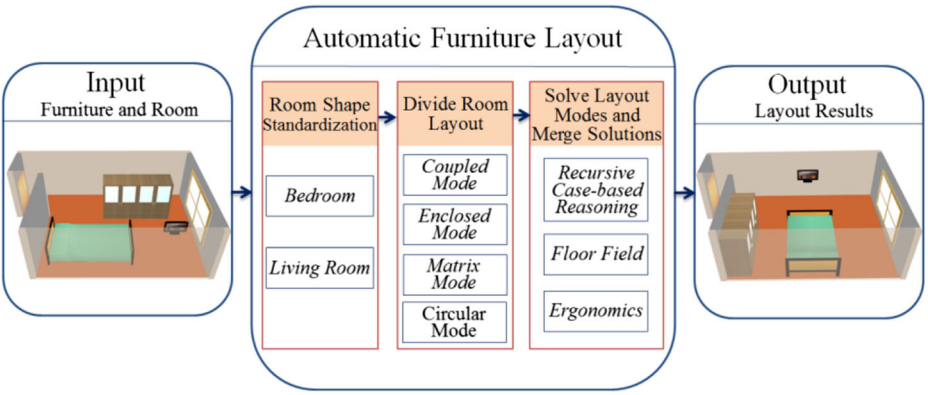


Fig. 2 The overview of our proposed system

requires that the split points are outside the polygon P. For example, the polygon P shown in Fig. 3a can generate nine split points shown in Fig. 3b using the above approach. The point p_1, p_2, p_3, p_4, p_5, p_6, p_7 and p_8 belong to the internal region split points, and the point p_9 belongs to the external region split points.

4.2 Bedroom shape standardization

Let the polygon $P = p_1p_2 \ldots p_n$ denote the floor shape of the bedroom. We convert the problem of bedroom shape standardization into the following problem: use the internal region split points of P to construct the rectangles whose area are greater than zero, and find the rectangle whose area is the largest in these rectangles. It can be expressed as follows:

$$Max\{Area[rect(v_1, v_2, v_3, v_4)]\} \tag{1}$$

where v_1, v_2, v_3 and v_4 denote the four vertices of the rectangle $rect(v_1, v, v_3, v_4)$ that are required within the polygon P, and $Area[\ rect(v_1, v, v_3, v_4)]$ denotes the area of rectangle $rect(v_1, v, v_3, v_4)$. The approach to solve the formula (1), namely bedroom shape standardization algorithm, is given as follows.

Algorithm 1 Bedroom shape standardization
Input: A bedroom floor shape $P = p_1p_2 \ldots p_n$
Output: The rectangular layout region

Step1: Initially, use an array A to save internal region split points, and use an array B to save rectangles.

Step2: Find all the internal region split points of P by the approach to generate region split points in Section 4.1, and add these points to A.

Step3: Construct all the rectangles whose areas are greater than zero using the points in A whose size is m, and use B to save these rectangles.

Step 3.1: FOR $i \leftarrow 1$ TO m
Step 3.2: FOR $j \leftarrow 1$ TO m
Step 3.3: IF i is not equal to j THEN
Step 3.4: Use two vertices $A[i]$ and $A[j]$ to construct a two dimensional axis-aligned bounding box, denoted by Box.
Step 3.5: IF the area of the Box is greater than zero and the four vertices of the Box are in P THEN
Step 3.6: Add the Box, namely a rectangle, to B
Step 3.7: END IF
Step 3.8: END IF
Step 3.9: END FOR
Step 3.10 END FOR

Step4: Find the rectangle of the largest area in B, and return the region that the rectangle contains, namely the rectangular region.

In this algorithm, the rectangles are constructed according to the two points in A. Step 3.3 and 3.5 ensure that the four vertices of the rectangle are within A and the area of the rectangle

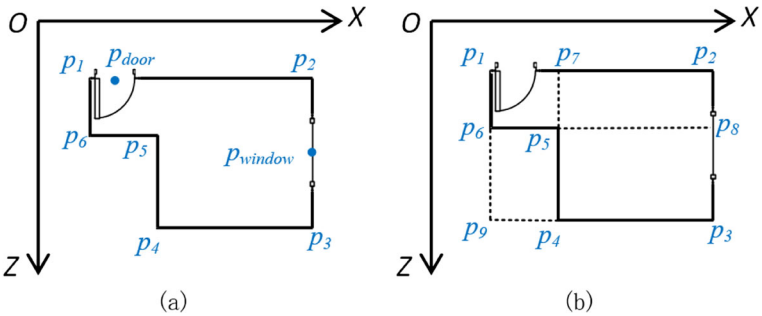


Fig. 3 **a** An example of the floor shape of the bedroom **b** The region split points of the floor shape of the bedroom

is greater than zero. To verify the correctness of this algorithm, we perform the algorithm on four bedroom floor shapes that are shown in Fig. 4. The shaded rectangular regions in Fig. 4a, b, c, d are extracted by the algorithm, which demonstrates that this algorithm is effective.

4.3 Living room shape standardization

Let the polygon $P = p_1p_2 \ldots p_n$ denote the floor shape of the living room. We assume that the two walls that the television and the sofa lean against are p_ap_b and p_cp_d (see Fig. 5). The problem of living room shape standardization can be converted into the following problem: use internal region split points of the polygon $Q = p_ap_bp_cp_d$ to construct the rectangles whose area are greater than zero, and find the rectangle whose area is the largest in these rectangles. It can be expressed as follows:

$$Max\{Area[rect(v_1, v_2, v_3, v_4)]\} \tag{2}$$

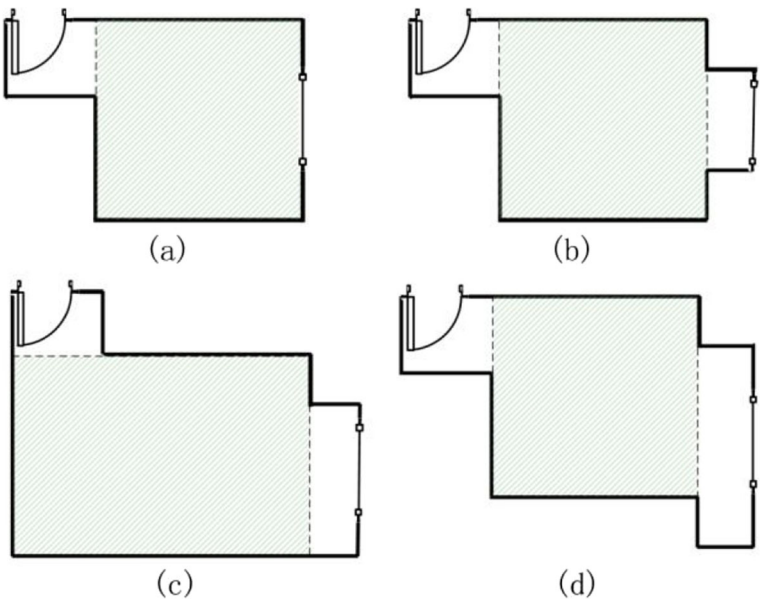


Fig. 4 Four examples of bedroom shape standardization

where v_1, v_2, v_3 and v_4 denote the four vertices of the rectangle $rect(v_1, v, v_3, v_4)$ that are required within the polygon Q, and $Area[\ rect(v_1, v, v_3, v_4)]$ denotes the area of rectangle $rect(v_1, v, v_3, v_4)$. The approach to solve the formula (2), namely living room shape standardization algorithm, is given as follows.

Algorithm 2 Living room shape standardization
Input: A living room floor shape $P = p_1p_2 \ldots p_n$
Output: The rectangular layout region

Step1: Initially, use an array A to save internal region split points, and use an array B to save rectangles; use an array C to save the edges of P; let $C[i]$ be the wall that the window leans against.

Step2: Find the two walls that the television and the sofa lean against, and require that the length of each wall is greater than d.

Step2.1: $j \leftarrow i-1,\ k \leftarrow k+1$;
Step2.2: WHILE the length of $C[j] < d$
Step2.3: $j \leftarrow j-1$;
Step2.4: END WHILE
Step2.5: WHILE the length of $C[k] < d$
Step2.6: $k \leftarrow k+1$;
Step2.7: END WHILE
Step2.8: $p_ap_b \leftarrow C[j], p_cp_d \leftarrow C[k]$;

Step3: Find the internal region split points of Q by the approach to generate region split points in Section 4.1, and use A to save these points.

Step4: Similar to Step 3 in the algorithm 1, construct all the rectangles whose areas are greater than zero using the points in A, and require that the four vertices of these rectangle rectangles are within P and Q. Add these rectangles to B.

Step5: Find the rectangle of the largest area in B, and return the region that the rectangle contains, namely the rectangular region.

In this algorithm, Step 2.1 and 2.7 ensure that the two walls, p_ap_b and p_cp_d, are longer than d. We set d =200, that is, the two walls are longer than 2 m. To verify the correctness of this algorithm, the algorithm performs on the three living room floor shapes shown in Fig. 5. The shaded rectangular regions in Fig. 5a and b are extracted by the algorithm, which demonstrates that this algorithm is effective. If the length of the rectangular region is smaller than the length of the sofa, then the region cannot be arranged the sofa. We could expand the region by increasing the length of the region. For example, in Fig. 5c, the length of the rectangular region is smaller than the length of the sofa. We can expand the region to the left, and the shaded rectangular region in Fig. 5d is an expanded region that can be arranged the sofa.

5 Solution for coupled mode

The coupled mode represents the layout relationship between the two types of furniture. The idea of solving the coupled mode is that solving new problems is according to the solutions of similar past problems. We use case-based reasoning to realize this idea. We first set up a mathematical model to represent the layout relationship, and use the

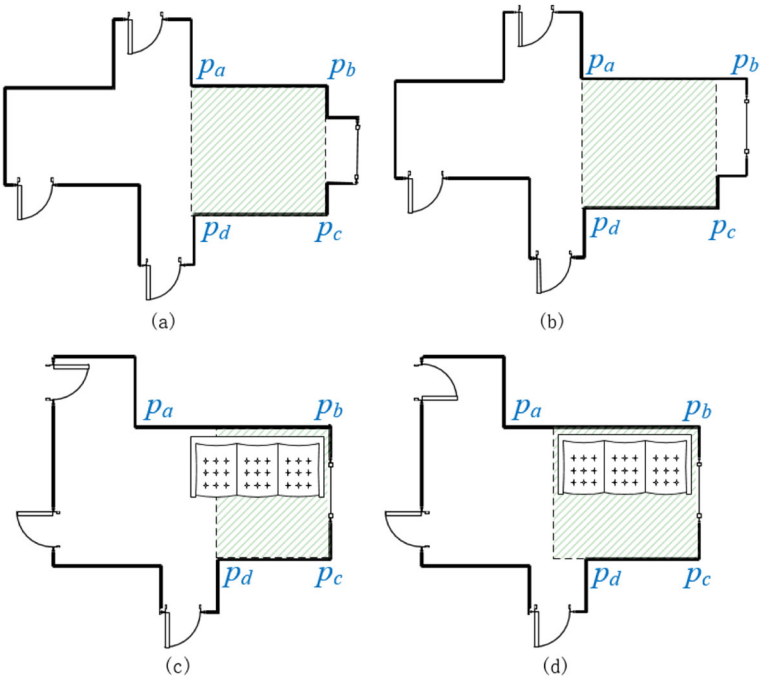


Fig. 5 **a** and **b** give two examples of living room shape standardization **c** and **d** illustrate the case of expanded the rectangular region

parameters of the mathematical model to describe the problem and solution. Then the nearest neighbor algorithm is used to retrieve cases. In addition, to solve the multilevel coupled relationship among the furniture, we construct the case tree and propose the algorithm of recursive case-based reasoning.

5.1 Mathematical model

We use a mathematical model to denote the layout relationship among the furniture in the coupled mode. A coupled mode consists of one piece of parent furniture and n pieces of identical child furniture. As shown in Fig. 6, the parent furniture is the desk, and the child furniture is the stool. The parent furniture contains n layout points, and each piece of child furniture has one layout point. We use the axis-aligned bounding box to represent a piece of furniture, and use k to denote the child furniture number, $1 \leq k \leq n$. Let l, w and h denote the length, width and height of the furniture, respectively. Let $S = (l/2, h/2, w/2)^T$ be the size matrix, S_0 be the size matrix of the parent furniture, and S_k be the size matrix of the child furniture k. WLet $P = (x, y, z)^T$denote the furniture position, P_0 denote the parent furniture position, and P_k denote the position of the child furniture k. Let a be the furniture rotation angle around its Y axis with $a = \{0,90,180,270\}$, a_0 be the child furniture rotation angle, and a_k be the rotation angle of the child furniture k. The main idea of the coupled mode algorithm is this: we find the layout points of the parent furniture and that of the child furniture and then obtain the layout results through these layout points and matrix transformation. The position relationship between the child furniture k and the parent furniture can be modeled as follows:

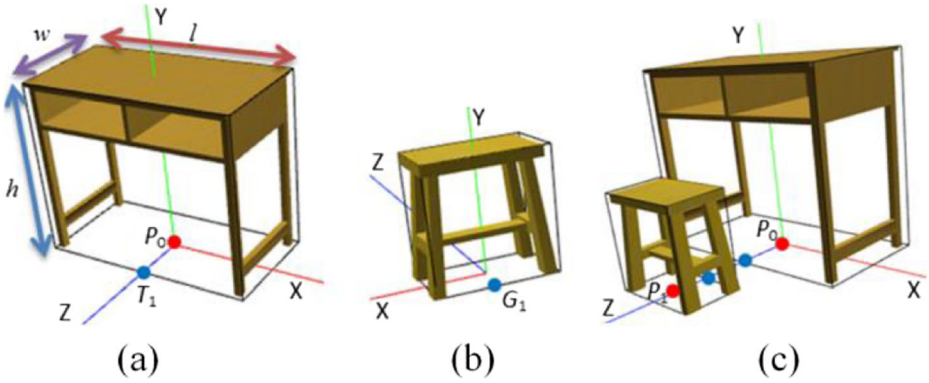


Fig. 6 **a** The information of the parent furniture desk **b** The information of the child furniture stool **c** The information of the coupled mode

$$\begin{aligned} Pk &= Gk + Tk + Dk \\ Gk &= MkSk \\ Tk &= P0 + RkS0 \end{aligned} \tag{3}$$

where $G = (g_{ij})_{3\times 1}$,and G_k denotes the layout point of the child furniture$k$;$T = (t_{ij})_{3\times 1}$, and T_k denotes the parent furniture layout point relative to the child furniture k; $D = (d_{ij})_{3\times 1}$,and $D_k$denotes the child furniture k distance matrix; $M = (m_{ij})_{3\times 1}$,and $M_k$denotes the child furniture k transformation matrix; $R = (r_{ij})_{3\times 1}$,and R_k denotes the parent furniture transformation matrix relative to the child furniture k; P_0,$S_0$and $S_k$are known. a_k,M_k, R_k and $D_k$are set according to ergonomics and the real layout case. To obtain D_k, we can consult the data of ergonomics or measure the distance among furniture in the real layout case. For two different name cases, we use the different distance matrix D_k, such as the case of the sofa and the tea table and the case of the table and the chair. For two same name cases, the distance matrix D_k of the two cases may be the same or different, and it depends on the size of the furniture in the case. For example, the distance matrix D_k of the case of the college student's desk and stool is different from that of the case of the children's desk and stool.

The parameters of the mathematical model of the desk and stool in Fig. 6 are as follows.

$$\mathrm{a0} = 0, \mathrm{a1} = 0$$
$$\mathrm{M_{01}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathrm{M_1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathrm{D_1} = \begin{bmatrix} 0 \\ 0 \\ 20 \end{bmatrix}$$

5.2 Furniture-furniture coupled mode algorithm

Case-based reasoning has been widely used to solve various problems [3, 12, 14], and we use it to solve the coupled mode. The layout case consists of two parts: the layout problem description and the layout solution description. The layout problem is described by the vector $q = (l_0, w_0, h_0, l_1, w_1, h_1)^T$, where l_0, w_0, h_0, l_1, w_1, and h_1 denote the length, width, and height of the parent and the child furniture, respectively. The layout solution is the parameters of mathematical model. The layout case name is composed of the parent furniture name, the child furniture name and the total amount of the child furniture. Case similarity is

defined as the Euclidean distance between the two layout cases. We use the nearest neighbor algorithm (K-Nearest Neighbor algorithm, K = 1) to retrieve cases.

$$\underset{i}{Min}\left\{Dis(case_0, case_i) = \sqrt{\sum_{j=1}^{6}\left(q_{0,j}-q_{i,j}\right)^2}\right\} \tag{4}$$

where $case_0$ denotes a await layout case and $case_i$ denotes a layout case identical to the name of $case_0$ with $1 \le i \le m$. The problem description of $case_0$ and the problem description of $case_i$ are denoted by $q_0$andq_i, respectively. The minimization returns the best case $case_{best}$ which is the most similar to $case_0$ in the case library. Substituting the solution of $case_{best}$ and known parameters P_0, $S_0$and S_k of the $case_0$ into the mathematical model, we can obtain a layout result. The algorithm for one coupled mode is given below.

Algorithm 3 Furniture - furniture coupled mode
Input: An await layout $case_0$
Output: Layout results of the $case_0$

Step 1: Find a set of case whose name is identical to the $case_0$ from the case library.
Step 2: Find the $case_{best}$ using the nearest neighbor algorithm using formula (4).
Step 3: Get the parameters from the mathematical model of the $case_{best}$.
Step 4: Assign the parameters to the mathematical model of $case_0$ and calculate using formula (3).
Step 5: Obtain and return the layout result of $case_0$.

5.3 Furniture-room coupled mode algorithm

When there is a coupled layout mode between furniture and a room, the room can be treated as furniture. Thus furniture-room coupled mode is converted into the furniture - furniture coupled mode. If the room shape is non-rectangular, then the rectangular region of the room should be extracted with room shape standardization algorithm in Section 4. The common furniture-room coupled modes are summarized as follows. (1) The tables are arranged in the middle of the conference room (see Fig. 7a). (2)The sofas are arranged along the windows of the living room (see Fig. 7b). (3) The bed is arranged along the windows of the bedroom (see Fig. 7c), and the distance D_{bed} is determined by the room size.

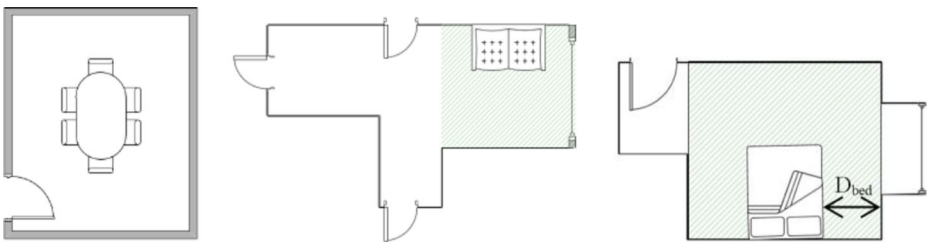


Fig. 7 **a** Meeting room **b** Non-rectangular living room **c** Non-rectangular bedroom

5.4 Recursive case-based reasoning

Note that one coupled mode can be solved using Algorithm 3. However, there may be more than one coupled mode among the furniture in a room. As shown in Fig. 8, there exist four coupled modes, namely, the table and chair, the table and computers, the table and book, and the computer and mouse. For this case, we use both recursive techniques and case-based reasoning to solve. The algorithm is given below.

Algorithm 4 Multiple coupled modes
Input: A case tree
Output: Layout results of the furniture in the tree.

Step 1: Construct a case tree by retrieving the case library.
Step 2: Use an array A to save the furniture that has been arranged.
Step 3: Perform the following procedure starting from the root node of the tree.
Recursive Case-based Reasoning (root)

Procedure：Recursive Case-based Reasoning(node)

Step 1: Let n denote the number of children of the node.
Step 2: IF n >0 THEN
Step 3: Sort children furniture of the node by their bounding box size
Step 4: FOR $i \leftarrow 1$ TO n
Step 5: child←children[i]
Step 6: Try to arrange the child using formula (3)
Step 7: IF no collision between the child and the furniture in A THEN
Step 8: Arrange the child using formula (3)
Step 9: Put the child into A
Step 10: Recursive Case-based Reasoning (child)
Step 11: ELSE
Step 12: Arrange the child on the support surface. If arrange failed, then the system advise the user to manual arrange the child furniture.
Step 13: END IF
Step 14: END FOR
Step 15: END IF

In Step 7 of this procedure, we use the AABB bounding box of the furniture to perform collision detection. In Step 12, the support surface refers to the floor of the room or the surface of parent furniture. In Step 6, if we calculate the Y axis coordinate position of the child furniture that equal to zero, then the support surface of the child furniture is floor; otherwise, the support surface of the child furniture is the surface of parent furniture. For example, the layout tree is shown in Fig.8 and the layout result is shown in Fig. 9b. The arranging furniture process using recursive case-based reasoning is as follows.

Step 1: Arrange the table, and find the child nodes of the table; Sort these children by size, and then obtain the following sequence: chairs, computers and books. Put the table into the array A.
Step 2: Arrange the chair and put it into A.

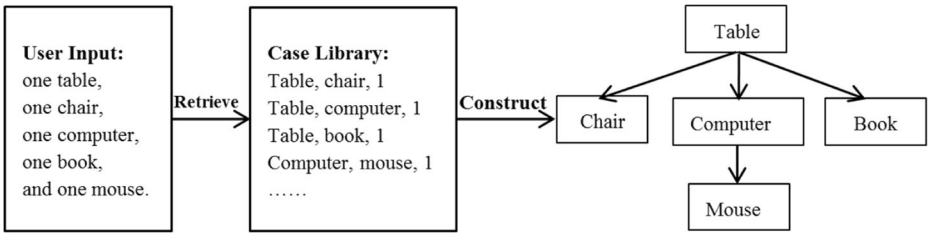


Fig. 8 Process of constructing a case tree

Step 3: Arrange the computer. The computer has a child of the mouse, and thus we should arrange the mouse. Similarly, put both the computer and the mouse into *A*.

Step 4: Arrange the book on the table. The book should be arranged in the same position of the computer according to the case library, which leads to the collision between the book and the computer (see Fig. 9a). Therefore we should find a position on the support surface of the table again to arrange the book, and finally the book is arranged in the upper left corner of the table (see Fig. 9b).

6 Solution for enclosed mode

The enclosed mode requires the furniture to be arranged along the wall. The key problem of enclosed mode is to determine the position of the furniture along the wall.

To ensure that the furniture is arranged along the wall, we first construct the attractive energy field for each wall. On the other hand, the layout of furniture along the wall cannot affect the walking. We construct the repulsive energy field for frequent paths in the room. Thus the furniture is arranged in a position where the furniture energy value is the largest, which ensure that the furniture is along the wall and normal walking.

6.1 Floor field

Many scholars used various floor fields to solve the problem of pedestrian evacuation [8, 11, 21]. This paper presents a floor field suitable for furniture layout to solve the

Fig. 9 **a** A collision between the book and the computer (**b**) The finally layout results

enclosed mode. The floor field guides furniture layout according to the energy distribution. Before constructing the floor field, the room**'s** floor should be discretized. We use 1 unit to represent 1 cm in the real world. For example, the length and width of the room are 4 m and 3 m, respectively, and then the size of the room's floor is expressed as 400*300. The room floor field consists of two parts: walls energy field and paths energy field. For example, a bedroom floor field is shown in Fig. 10a.

A wall energy field is an attractive energy for the furniture. Let t denote the maximum energy intensity of the wall, $t > 0$. Assume that the shortest distance from the point P(x, z)to the wall p_ip_j is d_1, and w_1 denotes the width of the wall energy field (see Fig. 10b). The energy value of the point P(x, z) inside the room generated by the wall $p_ip_j$is calculated as follows.

$$V\left(P(x,z), wall\left(p_ip_j\right)\right) = \begin{cases} \frac{d_1}{w_1} \times t, d_1 \leq w_1 \\ 0, d_1 > w_1 \end{cases} \tag{5}$$

A path energy field is a repulsive energy for the furniture. Let s denote the maximum energy intensity of the path, $s < 0$. Assume that the shortest distance from the point P(x, z) to the path $p_{door}p_{window}$ is d_2, and w_2 denotes the width of the path energy field (see Fig. 10c). The energy value of the point P(x, z) inside the room and on the path $p_{door}p_{window}$ is calculated as follows:

$$V(P(x,z), Path(i)) = \begin{cases} \frac{d_2}{w_2} \times s, d_2 \leq w_2 \\ 0, d_2 > w_2 \end{cases} \tag{6}$$

Given the positions of doors and that of the window, any point's energy value in the room can be calculated as follows:

$$V(P(x,z)) = \sum_{i=1}^{n} V(P(x,z), wall_i) + \sum_{j=1}^{m} V\left(P(x,z), path_j\right) \tag{7}$$

where n and m denote the number of walls and the number of paths, respectively. According to ergonomics guidelines, our parameter settings are as follows: $t = 50$, $s = -50$ and $w_1 = w_2 = 50$.

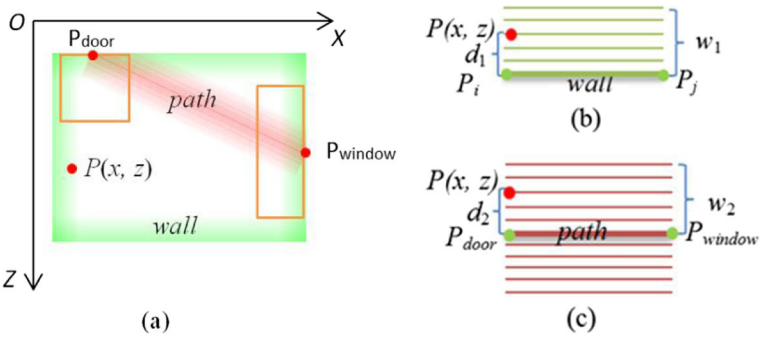


Fig. 10 **a** The floor field of the room (**b**) The energy field of the wall (**c**) The energy field of the path

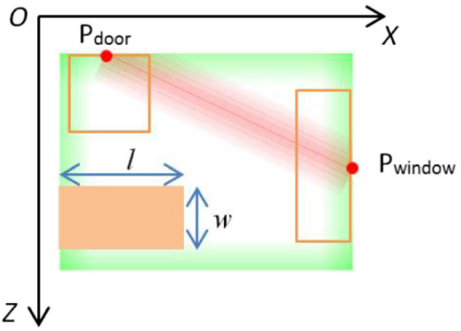


Fig. 11 Furniture energy

6.2 Furniture energy and furniture energy maximization

The furniture energy denotes the sum of the energy values of the points occupied by the furniture. As shown in Fig. 11, the rectangle is the bottom of the furniture bounding box, and the furniture energy is calculated as follows:

$$E(P(x,z),S(l,w)) = \sum_{i=x-l/2}^{x+l/2}\sum_{j=z-w/2}^{z+w/2} V(P(i,j)) \tag{8}$$

where point P(x, z) denotes the furniture position, and l and w denote the length and width of the furniture, respectively.

Furniture energy maximization refers to searching the position in the room where the furniture energy value is the largest, which is represented as follows:

$$\underset{x,z}{Max}\, E(P(x,z),S(l,w)) \tag{9}$$

where $E(P(x,z),S(l,w))$ denotes the energy value of furniture in position$P(x,z)$. The furniture can be rotated with 0, 90,180 or 270 degrees in the search process of energy maximization. To improve the computational efficiency, the energy value of the furniture in current position can be calculated according to the energy value of the furniture in the previous position. Note that if the furniture collides with other furniture or the door's region (see Fig. 11), and then the furniture cannot be arranged. In addition, in a bedroom, if the furniture collides with the window's region (see Fig. 11) and the furniture's height is higher than the specified height, and then the furniture will not be arranged. A comparison is made between the distance field in [27] and the floor field. Figure 12a shows the energy distribution using the distance field in [27], and Fig. 12b shows the energy distribution using our floor field. Figure 12c and d show the layout results generated using distance field in [27] and our floor field. The results show that our floor field is more suitable for furniture layout in a non- rectangular room.

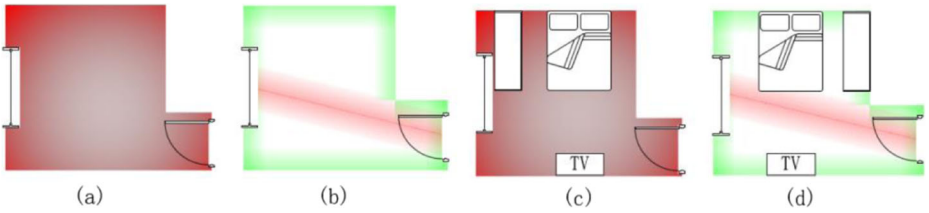


Fig.12 a The energy distribution using the distance field in [27] (**b**) The energy distribution using our floor field (**c**) The layout result using the distance field in [27] (**d**) The layout result using our floor field

7 Solution for matrix mode and circular mode

The layout of the matrix mode and that of the circular mode are relatively simple. When the distance between the furniture is determined, the layout result can be obtained according to geometric calculation. In this section, we use the way of retrieving case to obtain the distance between the furniture.

7.1 Solution for matrix mode

The matrix mode is similar to the regular interval placement proposed in [2]. Furniture grouping is the emphasis in [2]. Here we focus on how to obtain the distance relationship among the furniture through the case library. Note that all the furniture is identical in the matrix mode. As shown in Fig. 13, let *l, w* and *h* denote the length, width and height of the furniture, respectively. Let d_v and d_h denote the vertical distance and horizontal distance among furniture, respectively. We store the parameters, namely, the real case, the scene type, the furniture name, *l, w*, *h*, d_v and d_h, into the case library. The case name consists of the scene type and furniture name. The parameters *l, w*, *h*, d_v and d_h are used to describe the case.

Consider that there exists furniture of matrix mode to be arranged, and find the furniture layout results. Assume that the furniture should be arranged in a rectangular region, and the location and size of the rectangular region have been determined. The layout process of the matrix mode is as follows.

Sept 1: Find the cases with identical name in the case library.
Sept 2: Similar to algorithm 3, retrieve the best case from these identical name cases using the nearest neighbor algorithm.
Sept 3: Find the parameters d_v and d_h from the best case.
Sept 4: According to the parameters d_v and d_h and the rectangular region, we can obtain the layout results using geometry calculation. Here we omit the process of the geometry calculation since they are simple.

7.2 Solution for circular mode

Here we use an example to illustrate the circular mode algorithm. Given a round table and *n* chairs shown in Fig. 14, find the layout results among the furniture. Let *r* denote the radius of the round

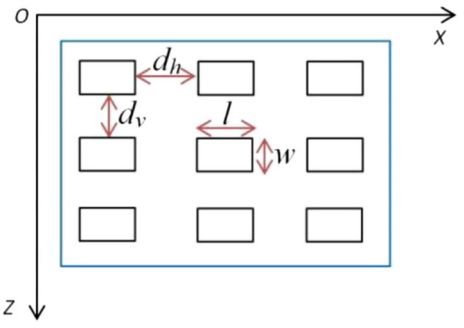


Fig. 13 Distance parameters of the matrix mode

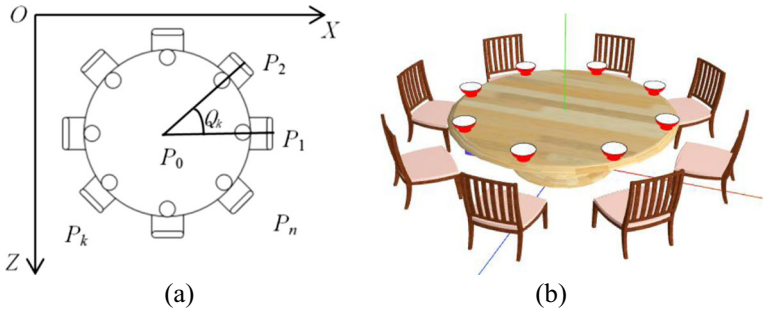


Fig. 14 An example of the circular mode

table. The length and width of the chair are equal, denoted by l. Let $P_0(x, y, z)$ denote the position of the round table and $P_i(x, y, z)$ denote the position of the kth chair, $1 \le k \le n$. Assume that the n chairs can be arranged next to the table, and these chairs have no collision with each other.

The position of the kth chair can be obtained using the following formula.

$$\begin{aligned} Pk(x) &= P0(x) + (r + l/2 + d) \times \cos(Qk) \\ Pk(z) &= P0(z) - (r + l/2 + d) \times \sin(Qk) \\ Pk(y) &= P0(y) \end{aligned} \tag{10}$$

where Q_k denotes the angle of the kth chair rotating around Y axis of the round table, $Q_k = (k - 1) \times 360/n$ (see Fig. 14a), d denotes the distance between the round table and the chair, and here d equals to zero. The distance d is required to meet the needs of ergonomics, and it also can be stored in the case library. The angle of the kth chair rotating around its Y axis equal to $\text{Angle}_k = Q_k - 90$. Similarly, we also can obtain the layout result between the round table and the bowls (see Fig. 14b).

8 Furniture layout algorithms for four scenes

In the previous sections, we proposed the general framework of furniture layout algorithms, and the solutions for the four layout modes. Based on the framework and these solutions, in

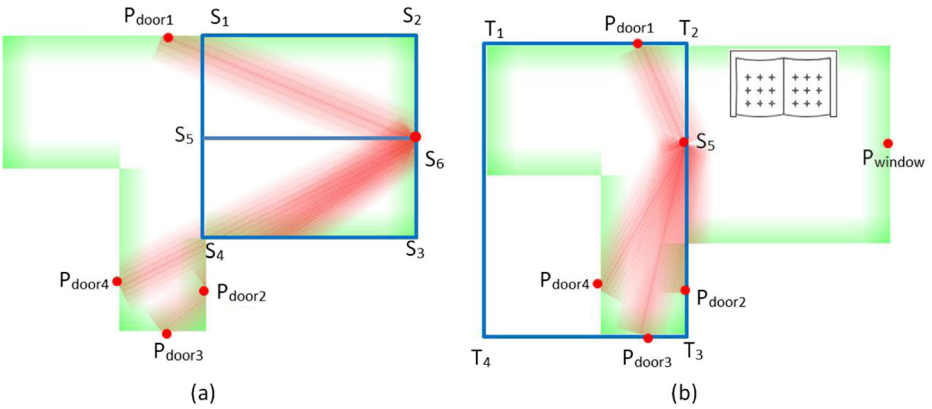


Fig. 15 **a** Energy distribution of finding layout region for the sofa (**b**) Energy distribution of finding the position of the dining table

this section we give the respective furniture layout algorithms for a living room, a bedroom, a classroom and a restaurant.

8.1 Living room furniture layout algorithm

We consider the following problem. Given a living room which contains furniture such as sofas, televisions, dining chairs, dining tables and so on, generate a reasonable layout for the living room. Figure 15a shows the energy distribution of finding the layout region for the sofa, and Fig. 15b shows the energy distribution of finding the position of the dining table. The furniture layout algorithm for the living room is given as follows.

Algorithm 5 Living room furniture layout
Input: A living room and furniture
Output: Furniture layout results of the living room.

Step1: Arrange the furniture which belongs to the coupled mode using Algorithm 4 in Section 5.4.

Step2: Extract the rectangular region $S_1S_2S_3S_4$ using the living room shape standardization algorithm in Section 4, and then divide the region $S_1S_2S_3S_4$ into two small regions: $S_1S_2S_6S_5$ and $S_5S_6S_3S_4$.

Step3: Generate floor field using formula (5), (6) and (7), which is shown in5. 14a. Calculate energy value for the two small regions using formula (8).

Step4: Arrange the television and sofa. The region with lower energy values is used to arrange the television, and the region with higher energy values is used to arrange the sofa. As shown in Fig. 15a, the region $S_1S_2S_6S_5$ and $S_5S_6S_3S_4$ are used to arrange the sofa and television, respectively.

Step5: Regenerate floor field, which is shown in Fig. 15b.

Step6: Arrange the dining table and make its energy maximization using formula (9). The position of the dining table can be searched within the region $T_1T_2T_3T_4$.

Step7: Sort the remaining furniture by size. Arrange the remainder furniture to make its energy maximization respectively. Return the layout results.

In the Step 2 of this algorithm, if the length of the region $S_1S_2S_6S_5$ is smaller than the length of the sofa, then the position of the sofa should be adjusted. In Step 5, as the position of the sofa has been found, we change the current end point of paths, point P_{window}, to the point S_5. Thus, the floor field needs to be regenerated.

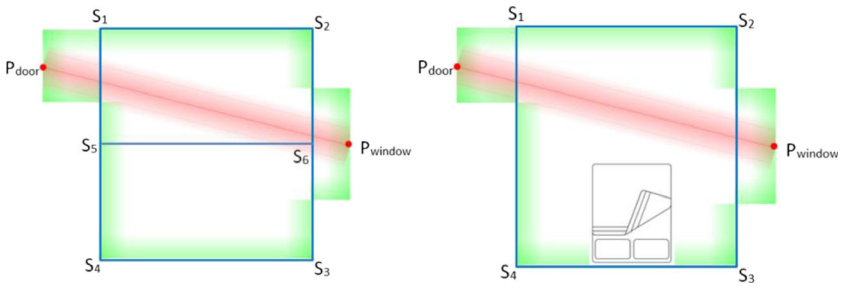


Fig. 16 **a** Energy distribution of the bedroom (**b**) Finding the position of the bed

8.2 Bedroom furniture layout algorithm

We consider the following problem. Given a bedroom which contains furniture such as beds, televisions, cabinets and so on, generate a reasonable layout for the bedroom. Figure 16a shows the energy distribution of the bedroom, and Fig. 16b shows the position of the bed.The furniture layout algorithm for the bedroom is given as follows.

Algorithm 6 Bedroom furniture layout
Input: A bedroom and furniture
Output: Furniture layout results of the bed room.

Step1: Arrange the furniture which belongs to the coupled mode using Algorithm 4 in Section 5.4.

Step2: Extract the rectangular region $S_1S_2S_3S_4$ using the bedroom room shape standardization algorithm in Section 4, and then divide the region $S_1S_2S_3S_4$ into two small regions: $S_1S_2S_6S_5$ and $S_5S_6S_3S_4$.

Step3: Generate the floor field using formula (5), (6) and (7), which is shown in Fig. 16a. Calculate energy value for the two small regions using formula (8).

Step4: Arrange the television and the bed. The region with lower energy values is used to arrange the television, and the region with higher energy values is used to arrange the bed. As shown in Fig. 16a, the region $S_1S_2S_6S_5$ and $S_5S_6S_3S_4$ are used to arrange the television and the bed, respectively.

Step5: Sort the remaining furniture by size. Respectively arrange the remainder furniture to make its energy maximization using formula (9). Return the layout results.

8.3 Classroom furniture layout algorithm

We consider the following problem. Given a classroom which contains furniture such as a teacher's desk, students' desks, and students' chairs and so on, generate a reasonable layout for the classroom. Assume that the floor shape of the classroom is rectangular. The furniture layout algorithm for the bedroom is given as follows.

Algorithm 7 Classroom furniture layout
Input: A classroom and furniture
Output: Furniture layout results of the classroom

Step1: Arrange the students' desks and chairs using Algorithm 4 in Section 5.4.

Step 2. Arrange the students' desks using the algorithm for matrix mode in Section 7.1.

Step 3. Arrange the teacher's desk using furniture-room coupled mode algorithm in Section 5.3.

Step 4: Return the layout results.

8.4 Restaurant furniture layout algorithm

We consider the following problem. Given a restaurant which contains furniture such as a round table, chairs and so on, generate a reasonable layout for the restaurant. Assume that the

floor shape of the restaurant is rectangular. The furniture layout algorithm for the restaurant is given as follows.

Algorithm 8 Restaurant furniture layout
Input: A restaurant and furniture
Output: Furniture layout results of the restaurant room.

Step1: Arrange the furniture which belongs to the coupled mode using Algorithm 4 in Section 5.4.

Step2: Arrange the round tables and chairs using the algorithm for circular mode in Section 7.2.

Step3: Arrange the round tables using the algorithm for matrix mode in Section 7.1.

Step4: Return the layout results.

9 Evaluation method for layout results

An objective and convenient layout evaluation method for layout results is an important part of the furniture layout algorithm. The evaluation methods are used to evaluate the effect of the layout algorithms and compare performance among these furniture layout algorithms. It is not easy to evaluate the layout results, especially for the different requirements of many users and layout scenes. Many layout evaluation methods are based on layout rules [1, 5, 17, 24, 32]. These methods define an evaluation function by quantifying layout rules, obtaining the values of the function corresponding to the layout results, and then evaluate the layout results according to the values of the function. However, evaluation methods based on layout rules can hardly reflect users' evaluation for layout results. Since the layout results are eventually used by the users, it is appropriate for the users to evaluate the results. Hence, we propose a valuation method for layout results that is based on user feedback.

Assume that A pieces of furniture need to be arranged in a room. A furniture layout result of the room is generated using a furniture layout algorithm, and then a user's feedback information indicates that the positions of B pieces of furniture are wrong in the layout result, with $0 \leq B \leq A$. We use layout accuracy rate (LA) to represent the user's evaluation for the layout result, and layout accuracy rate is defined as follows:

$$LA = 1\text{-}\frac{B}{A} \tag{11}$$

where $0 \leq \mathrm{LA} \leq 1$, and the greater the value of $\mathrm{L}A$, the better the effect of the layout result. The two ways to obtain a user's feedback for a layout result are given below.

(1) The user's feedback information is obtained by online method. Assume that a user uses a 2D or 3D furniture design platform to arrange furniture. After a layout result has been generated using a furniture layout algorithm on the platform, the user moves the positions of B pieces of furniture in the layout result. We believe that the user is not satisfied with the layout result of the B pieces of furniture, namely, the user's feedback information indicates that the layout result of the B pieces of furniture is wrong. Determining whether the user has moved a piece of furniture can be achieved by detecting whether the position

of the piece of furniture has changed. The method of obtaining the user feedback online can be embedded in the 2D or 3D furniture layout platforms.

(2) The user's feedback information is obtained by off-line method. Assume that a user obtains the 2D rendering image of the layout result. We conduct a questionnaire on the user, and we ask the user which pieces of furniture should be rearranged again. If the user points out that the B pieces of furniture in the image need to be arranged again, then we believe that the user is not satisfied with the layout result of the B pieces of furniture, namely, the user's feedback information indicates that the layout result of the B pieces of furniture is wrong.

In addition, we also define average layout accuracy rate (ALA). Assume that a layout result is evaluated by N users, namely, there are N layout accuracy rates $\{LA_1, LA_2, LA_3, \ldots LA_N\}$ for the layout result. The average layout accuracy of the layout result is defined as follows:

$$ALA = \frac{\sum_{i=1}^{N} LA_i}{N} \tag{12}$$

where $0 \le LA \le 1$, and the greater the value of ALA, the better the effect of the layout result.

10 Experimental results and discussion

To validate efficiency of our algorithms, we conduct various experiments in our furniture layout system with the real interior design cases. We designed 16 common cases of the furniture layout to store in database of SQL Server 2012. The computation was performed on a computer with a processor of Dual-core 2.6GHz, main memory 4G, operation system Microsoft Windows 8 64bit and Chrome browser.

10.1 Comparisons with prior art

To evaluate our algorithms, we compared our algorithms with Liu's algorithms [17]. Since there is no size information for the two living rooms in literature [17], we manually extracted furniture and scene information from literature [17]. Then we conducted the two living room layouts using our algorithms with the extracted furniture and scene information. Figure 17 and Table 2 show the layout results and the detailed running time. The running time of our algorithms includes loading times of model files. As can be seen from Table 2, the computation efficiency of our algorithms is significantly higher than that of Liu's algorithms [17]. At the same time, we invited 10 volunteers to evaluate the four layout results. We obtained these volunteers' feedback information using the off-line method proposed in Section 9. For Fig. 17c, some of these volunteers deemed that the television cabinet was more suitable in the middle of the TV wall, and that the short sofa should be arranged by the window. For Fig. 17b and d, some of these volunteers deemed that it was better to leave a little space between the dining table and the wall. The four layout results' average layout accuracy rate (ALA) was computed according to formula 12. The average value of Fig. 17a's ALA and Fig. 17c's ALA is 0.90, and the average value of Fig. 17b's ALA and Fig. 17d's ALA is 0.92. The results show that the layouts of the two living rooms using our algorithms are reasonable and efficient.

10.2 Comparison with real designs

Given two real design cases shown in Fig. 18a and c, we also manually extracted furniture and scene information from the two cases. We invited an interior designer to manually design two layout results according to the extracted information on our system. The designer was allowed to manually drag an object and move it along the floor plane, but the designer was not allowed to use any algorithmic inference. It took him almost an hour to finish the classroom and the bedroom design. We conducted the bedroom and the classroom layout using our layout algorithms on the system. The running time of the bedroom furniture layout is 1.7 s, and the running time of the classroom furniture layout is 0.6 s. The layout results of the bedroom and the classroom using our algorithms are shown in Fig. 18b and d, respectively. The layout results of our algorithms are consistent with that of the two real design cases.

In addition, we conducted a restaurant layout experiment on our system. The restaurant contains 153 objects. The running time of the restaurant layout is 3.1 s. The running time also includes loading times of model files. The layout results of the restaurant using our algorithms are shown in Fig.19. Table 3 lists the layout modes that these experiment scenes contain.

Similarly, we also invited 10 volunteers to evaluate the three layout results using the off-line method proposed in Section 9. Some of these volunteers deemed that the two books in the

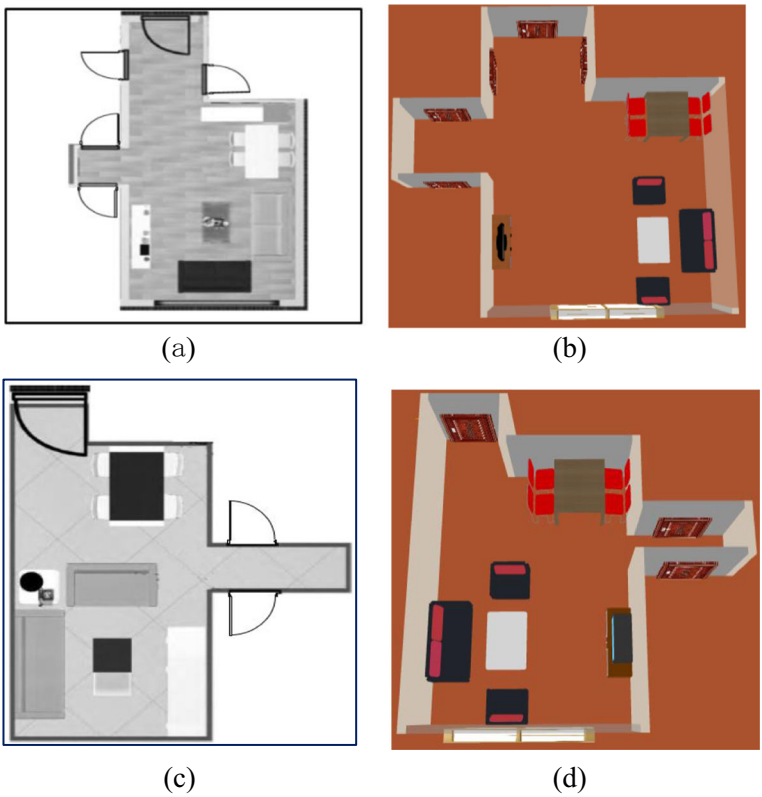


Fig. 17 **a** The layout result of living room 1 using Liu's algorithms [17] (**b**) The layout result of living room 1 using our algorithms (**c**) The layout result of living room 2 using Liu's algorithms [17] (**d**) The layout result of living room 2 using our algorithms

Table 2 Statistics of the layout results of the two living rooms

Algorithms	Computer configuration	Programming tools	Average running time (s)	Average value of two layout results' *ALA*
Our algorithms	RAM 4GB, and 2.6GHz CPU Intel Core i5-3230	WebGL and VS2015	1.5	0.92
Liu's algorithms [17]	RAM 4GB, and 3.4GHz CPU Intel Core i7-2600	Matlab and VS2008	50.3	0.90

bedroom were more suitable on the dresser, and the classroom platform should be not in the middle as it may block the sight of some students. By formula 12, the average layout accuracy rate of the bedroom, the classroom, and the restaurant are 0.87, 0.98, and 0.97, respectively.

10.3 Discussion

In Section 10.1 and 10.2, our algorithms are performed in the living rooms, the bedroom, the classroom and the restaurant. The experimental results are analyzed as follows.

(1) Results show that our algorithms can meet the functional requirement of these scenes. On the one hand, the techniques of case-based reasoning make layout results learn from the real cases, thus the results are reasonable. On the other hand, the layout rule generated by the floor field agrees with the real design rules. For non-rectangular living rooms and bedrooms, the paths in these rooms are found. The paths and walls in the room are given negative and positive energy, establishing a floor field, and then the furniture is arranged on the location with the largest energy value.

(2) The running time of our algorithms is suitable for real-time application. We divide the whole room layout into four furniture layout modes using the idea of divide and conquer,

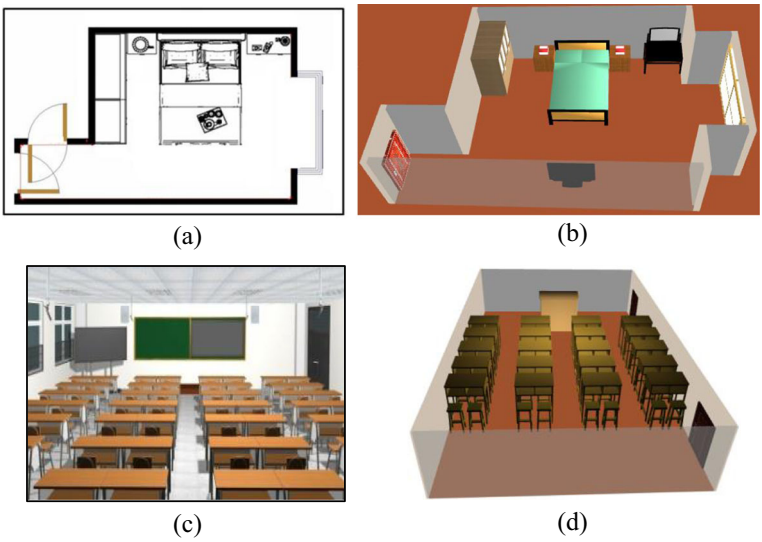


Fig. 18 **a** A real design case of the bedroom (**b**) The bedroom layout results using our algorithms (**c**) A real design case of the classroom (**d**) The classroom layouts result using our algorithms

Fig. 19 The layout results of the restaurant using our algorithms

reducing complexity of the room layout, and thus our algorithms are fast. At the same time, we can see that the running time of Liu's algorithms [17] is 50.3 s, which is difficult to meet real-time requirements of the online layout. Liu et al. [17] use the genetic algorithm to solve the furniture layout problem. The advantage of genetic algorithm is to be able to jump out from the local optimal. However, for a layout problem containing a lot of furniture, the layout relationships among the furniture are complicated, and the running time of the algorithm becomes longer. A similar problem is also found in [32]. Yu et al. [32] spend 219 s to deal with a restaurant layout problem using simulated annealing, and the restaurant contains 54 objects.

Based on the above analysis, we can conclude that our algorithms are effective, and can meet the needs of online real-time response. Our algorithms can be used in the online or off-line interior design platform to help users quickly create design drawing. In addition, another important application of our algorithms is modeling of 3D game scenes. In this scenario, our algorithms can be a function of the tool for automatically arranging 3D objects, which reduces users' workload.

The limitations of this work should be acknowledged. For one thing, our algorithms are not suitable for the indoor scenes with multiple functions. For example, assume that a room is used as both the living room and the bedroom, and the room contains furniture such as beds, televisions, cabinets, sofas, dining tables and so on. For this kind of rooms, our algorithms cannot obtain reasonable layout results. In practice, we found that this issue has little impact as

Table 3 The layout modes included in experiment scenes

Scene	Layout modes
Living room	Coupled mode: the sofas & the coffee table, the dining table & chairs, the living room & sofas, the sofas & the television cabinet, the television cabinet & the television
	Enclosed mode: the sofas, the dining table
Bedroom	Coupled mode: the bed & bedside cabinets, the bedside cabinets & books, the bedroom & the bed
	Enclosed mode: the bed, the cabinet, the dressing table
Classroom	Coupled mode: the students' desk & students' chair, the classroom & the teacher's desk, the teacher's desk & the students' desk
	Matrix mode: the students' desks
Restaurant	Matrix mode: the round table
	Circular mode: the round table & chairs, the round table & bowls

most indoor room functions are independent for each other. For another, our algorithms require that the walls of the room are parallel or vertical to each other. Except for in circular mode, our algorithms require that the furniture can only be rotated with 0, 90,180 or 270 degrees.

The future work can focus on the overall layout of the room which contains the light layout [25], color matching, and style matching [15]. Our work does not consider the two cases that the room is round or pentagonal and the room has curved walls or pillars. It would be significant to design algorithms to deal with the two cases in the future. At the same time, this paper only considers a room furniture layout, and this work can be extended to a larger scene, such as the shopping malls, the whole floor, or even the whole building.

11 Conclusion

In this paper, we have proposed online automatic furniture layout algorithms and developed a Web3D-based System based on these algorithms. A key component of our algorithms is that we solve the problem by dividing the whole scene layout into four layout modes, which reduce the layout complexity. To solve these layout modes and make the layout results reasonable, we utilize ergonomics guidelines, recursive techniques, case-based reasoning and floor field. We also present the room shape standardization techniques to handle the layout problem of the non-rectangular room and the evaluation method for layout results based on user feedback. Many experiments have been conducted in our layout system. The experimental results show that our algorithms are fast and effective, and the layout result of our algorithms can meet the functional requirement of the indoor scenes layout.

Acknowledgments The authors appreciate the comments and suggestions of all anonymous reviewers, whose comments significantly improved this paper. This work is supported by The Key Research Projects of Central University of Basic Scientific Research Funds for Cross Cooperation (201510-02), Research Fund for the Doctoral Program of Higher Education of China (No.2013007211-0035), National Nature Science Foundation of China (No. 61502306), National Nature Science Foundation of China (No. 61741203), and Key project in scientific and technological of Jilin Province in China (No.20140204088GX).

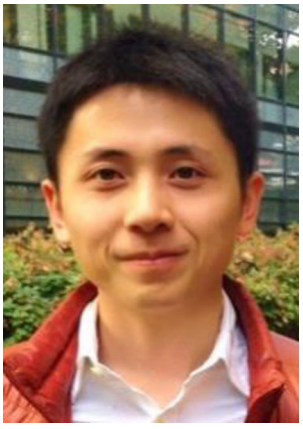
Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

1. Akase R, Okada Y (2013) Automatic 3D Furniture Layout Based on Interactive Evolutionary Computation. In Proc. Conf. on 2013 Seventh International Conference on Complex, Intelligent, and Software Intensive Systems, pp. 726-731
2. Akazawa Y, Okada Y, Niijima K (2002) Automatic 3d scene generation based on contact constraints. In Proc. Conf. on Eighth International Conference on Computer Graphics and Artificial Intelligence, pp 593-598
3. Besbes G, Baazaoui-Zghal H (2015) Modular ontologies and CRB-based hybrid system for web information retrieval. Multimedia Tools and Applications 74(18):8053–8077. https://doi.org/10.1007/s11042-014-2041-z
4. Bukowski RW (1995) Object associations: a simple and practical approach to virtual 3D manipulation. In proceedings of the Symposium on Interactive 3d Graphics, ACM, pp 131-138. doi: https://doi.org/10.1145/199404.199427
5. Chen G, Li G, Liu P, Ye T, Xian C (2014) Hierarchical constraints with particle swarm optimization for furniture arrangement. Journal of Computer-Aided Design and Computer Graphics 26(10):1603–1612
6. Esch G, Wonka P, Zhang E (2007) Interactive procedural street modeling. ACM Trans Graph 27(3):35:1–35:8. https://doi.org/10.1145/1278780.1278822
7. Fisher M, Ritchie D, Savva M et al (2012) Example-based synthesis of 3D object arrangements. ACM Trans Graph 31(6):135:1–135:15. https://doi.org/10.1145/2366145.2366154

8. Fu Z, Xia L, Yang H et al (2017) Simulation study of overtaking in pedestrian flow using floor field cellular automaton model. International Journal of Modern Physics C 28(5):1750059:1–1750059:17. https://doi.org/10.1142/S0129183117500590
9. Germer T, Schwarz M (2009) Procedural arrangement of furniture for real-time walkthroughs. Computer Graphics Forum 28(8):2068–2078. https://doi.org/10.1111/j.1467-8659.2009.01351.x
10. Guimarães MDP, Dias DRC, Mota JH et al (2016) Immersive and interactive virtual reality applications based on 3d web browsers. Multimedia Tools and Applications 75:1–15. https://doi.org/10.1007/s11042-016-4256-7
11. Huang HJ, Guo RY (2008) Static floor field and exit choice for pedestrian evacuation in rooms with internal obstacles and multiple exits. Physical Review E Statistical Nonlinear & Soft Matter Physics 78(1):021131:1–021131:6. https://doi.org/10.1103/PhysRevE.78.021131
12. Jin B, Xu S, Geng W (2016) Learning to sketch human facial portraits using personal styles by case-based reasoning. Multimedia Tools and Applications:1–25. https://doi.org/10.1007/s11042-017-4457-8
13. Kjølaas KAH (2000) Automatic Furniture Population of Large Architectural Models. Master's thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Cambridge, MA
14. Kolodner JL (1992) An introduction to case-based reasoning. Artif Intell Rev 6(1):3–34. https://doi.org/10.1007/BF00155578
15. Li W, Li W, Li W, Funkhouser T (2015) Style compatibility for 3d furniture models. ACM Trans Graph 34(4):85:1–85:9. https://doi.org/10.1145/2766898
16. Liu Y, Ch Z (2013) Ergonomics and interior design. China Electric Power Press, Beijing
17. Liu M, Jiang H, Mao TL, Wang Z (2016) Composite model for home furnishing generation. Chinese Journal of Computers 39(2):2:1–2:14. https://doi.org/10.11897/SP.J.1016.2017.02533
18. Merrell P, Schkufza E, Koltun V (2010) Computer-generated residential building layouts. ACM Trans Graph 29(6):1–12. https://doi.org/10.1145/1866158.1866203
19. Merrell P, Schkufza E, Li Z, Agrawala M, Koltun V (2011) Interactive furniture layout using interior design guidelines. ACM Trans Graph 30(4):87:1–87:9. https://doi.org/10.1145/2010324.1964982
20. Muller P (2006) Procedural modeling of cities. In Proc. Conf. on SIGGRAPH 2006, ACM, pp. 139-184.doi: https://doi.org/10.1145/1185657.1185716
21. Nishinari K, Kirchner A, Namazi A, Schadschneider A (2004) Extended floor field CA model for evacuation dynamics. IEICE Trans Inf Syst 87(3):726–732
22. Quax P, Liesenborgs J, Barzan A et al (2016) Remote rendering solutions using web technologies. Multimedia Tools and Applications 75(8):4383–4410. https://doi.org/10.1007/s11042-015-2481-0
23. Roux M L O L, Gaildrat V (2004) Using Meta-Heuristics for Constraint-Based 3D Objects Layout. In Proc. Conf. on Computer Graphics & Artificial Intelligence
24. Sanchez S, Roux O L, Gaildrat V et al. (2003) Constraint-based 3d-object layout using a genetic algorithm. In Proc. Conf. on Computer Graphics and Artificial Intelligence
25. Schwarz M, Wonka P (2014) Procedural Design of Exterior Lighting for buildings with complex constraints. ACM Trans Graph 33(5):166:1–166:16. https://doi.org/10.1145/2629573
26. Smith G, Salzman T, Stuerzlinger W (2001) 3D scene manipulation with 2D devices and constraints. In Proc. Conf. on Graphics Interface ,pp. 135-142
27. Song PH, Jia JY (2016) Online furniture layout method based on case-based reasoning and distance fields. Journal of System Simulation 28(10):2438–2447. https://doi.org/10.16182/j.cnki.joss.2016.10.021
28. Ulmer A, Müller P, Gool LV et al (2006) Procedural modeling of buildings. ACM Trans Graph 25(3):614–623. https://doi.org/10.1145/1141911.1141931
29. Xu K, Stewart J, Fiume E (2002) Constraint-based automatic placement for scene composition. In Proc. Conf. on Graphics Interface, pp 25-34
30. Xu K, Chen K, Fu H, Sun WL, Hu SM (2013) Sketch2Scene:sketch-based co-retrieval and co-placement of 3D models. ACM Trans Graph 32(4):123: 1–123:15. https://doi.org/10.1145/2461912.2461968
31. Xu W, Wang B, Yan DM (2015) Wall grid structure for interior scene synthesis. Comput Graph 46(1):231–243. https://doi.org/10.1016/j.cag.2014.09.032
32. Yu LF, Yeung SK, Tang CK, Terzopoulos D, Chan TF, Osher SJ (2011) Make it home: automatic optimization of furniture arrangement. ACM Trans Graph 30(4):86:1–86:11. https://doi.org/10.1145/2010324.1964981
33. Zorrilla M, Martin A, Sanchez JR et al (2014) HTML5-based system for interoperable 3D digital home applications. Multimedia Tools and Applications 71(2):533–553. https://doi.org/10.1007/s11042-013-1516-7

Peihua Song received the B.S. degree in Materials science and Engineering from Shandong University of Science and Technology, Shandong, China, in 2004, and the M.S. degree in Computer Science and Technology from Guangxi Normal University, Guangxi, China, in 2007. In 2007, he joined Guangxi Teachers Education University, Guangxi, China, as a lecturer. From 2014, he became a Ph.D. candidate at the School of Software Engineering, Tongji University, Shanghai, China. His current research interests include virtual reality, computer graphics and computer optimization technology.



Youyi Zheng received the B.S. and M.S. degrees in Mathematics from Zhejiang University, Hangzhou, China, in 2005 and 2007, respectively, and the Ph.D. degree in computer science from Hong Kong University of Science and Technology, Hong Kong, in 2011.From 2011 to 2015, he was a Postdoctoral Associate at the Department of Computer Science, Yale University and a Postdoctoral Fellow at KAUST. From 2015 to 2017, he was an Assistant Professor at ShanghaiTech University, Shanghai, China. In 2017, he joined State Key Lab of CAD&CG, College of Computer Science, Zhejiang University, Hangzhou, China, as a 100-talents plan researcher (PI). His current research interests include computer graphics, computer vision, and human-computer interaction.

Jinyuan Jia received the M.S. degrees in Mathematics from Jilin University, Changchun, China, in 1991, and the Ph.D. degree in computer science from Hong Kong University of Science and Technology, Hong Kong, in 2004. From 2004 to 2007, he was a professor in Zhuhai College of Jilin University, Zhuhai, China. In 2007, he joined the School of Software Engineering, Tongji University, Shanghai, China, as a Full Professor. His current research interests include computer graphics, CAD, geometric modeling, Web3D and computer simulation.

Yan Gao received the B.S. degree in Computer Software Engineering from Xi'dian University, Xi'an, China, in 2010, and the M.S. degree in Computer Software Engineering from Tongji University, Shanghai, China, in 2013. He is now a Ph.D. candidate at the Electrical and Computer Engineering department, Wallace H. Coulter School of Engineering, Clarkson University, Potsdam, New York, U.S. His current research interests include machine learning, data mining, software evolution, smart housing, and computer graphics technology.