Design and Implementation of Scenic Spot Recommendation System Based on Unity3D

Langcai Cao, Biyang Ma, Peiqiang Zeng, Bilian Chen*
Dept. of Automation Xiamen University
Xiamen, China
Email: blchen@xmu.edu.cn

Yinghui Pan*
Dept. of Information Management
Jiangxi university of Finance and Economics
Nanchang, China

***Abstract*—The function of traditional recommendation system is single, which lacks personalized service and virtual experience. This paper deals with the defection by scenic spot recommendation system based on Unity3D. We take Xiamen University and its surrounding attractions as an example to discuss the key technology of virtual scenic spots system based on Unity3D, including 3D modeling, building virtual scene, designing UI system, character interaction and recommendation algorithm. The recommendation system has two parts, one is the platform construction using Unity3D, where C# is the development language, and the other one is the algorithm designing that includes popular scenic spot recommendation algorithm and personalized scenic spot recommendation algorithm. Experimental results show that the routes recommended by the system have diverse scenic spots and meet users' preferences.**

Keywords-Unity3D; Platform Construction; Scenic Spot Recommendation Algorithm

I. INTRODUCTION

With the rapid development of tourism industry, the traditional service industry cannot meet the demand. The upgrading of tourism services will improve and transform traditional service industries into digital services with the help of "Internet Plus" technology. There are still weaknesses in scenic spot recommendation technology. For example, web-based scenic spot recommendation system is unable to show the spot features and satisfy the requirement of personalized for each user, electronic navigation system is shorted in interaction, individuation and experience in advance while can be remedied by the recommendation system with 3D virtual technology.

In this paper, we construct the recommendation system of the Xiamen University and the surrounding attractions, including the layout, campus teaching, life and traffic. It realizes the consistency of virtual and real environment, using 3D virtual technology, and makes a user feels like in a visual, auditory, and tactile real world. In the process, combining the 3D virtual technology and route recommendation algorithm is the highlight in this work, which brings a better visual experience and gives attractions route planning for a user. The personalized recommendation algorithm meets the personalized demand. However, it is not easy in designing personalized route recommendation algorithm to meet the demand of different persons and constructing the platform requires a great deal of time, especially in spot modeling.

II. DESIGN AND IMPLEMENTATION OF SYSTEM

A. The Overall Framework of the System

As shown in Figure 1, the overall framework of the system includes the user presentation layer, the business logic layer, and the data manipulation layer.

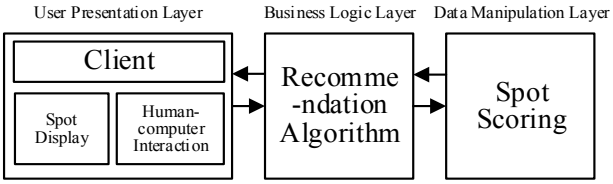

Figure 1 . The Overall Framework of the System.

The user presentation layer, consisting of the UI interface, route selection interface, user preference filling interface and scenic tour, is user-oriented. The business logic layer receives the data from the user presentation layer, and determines a route recommendation algorithm for a user, which includes popular scenic spot recommendation algorithm and personalized spot recommendation algorithm. The data manipulation layer is used to store the value of the evaluation of scenic spots, and the user's scoring value of the scenic spot is transmitted to the data manipulation layer every time.

B. System Development Process

Figure 2 shows the system development process. First, we need to obtain the data of models, including the CAD floor plan, figure, color, map, etc, and then import the data to the SketchUp and 3Dmax. Second, modeling and exporting the modeling to the Unity3D in Fbx format. Finally, we design UI interface and interactive link in the Unity3D, at the same time, route recommendation algorithm of popular spots and personalized recommendation algorithm are designed.

III. 3D MODELING

We use the Baidu map for latitude-longitude and skeleton map of a building, get its size by a measuring instrument and use the high-definition camera to obtain its characteristic patterns and chartlet before modeling. Then

978-1-5386-1829-5/17/$31.00©2017 IEEE

we organize the data for each building into a folder and use the Photoshop to

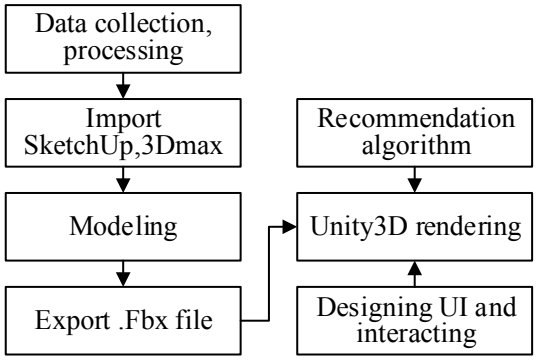

Figure 2 . System Development Process

change the map whose size is into standard form so that they can be completed stitching on all sides and have moderate brightness.

There are several modeling methods for a building, including GIS-based modeling, geometric modeling, image modeling, BIM technology modeling, etc [1,2]. This paper adopts image-based modeling method due to the fact that it is fast in modeling and easy to master.

The principle of image-based modeling method is the conversion from 2D to 3D and drawing the 3D model based on the characteristic parts form frontal, profile and top-viewed photos. And in the 3D modeling, it is necessary to build a new 3D coordinate system in the photos which is different from the 3D real world with x, y, z coordinate system. The modeling has four steps, as shown in Figure 3:

- Getting the characteristic parts of the building, and make sure that outlines of the key parts are clearly visible, and then importing it to the modeling software to adjust its size and location.
- Marking three-color positioning lines in the image. Red, green and blue are corresponding to x, y and z axis respectively. The red and green lever is used to adjust and match the perspective relationship in the picture. Building a three-dimensional coordinate system in the photo will facilitate subsequent mapping.
- Using the pen and curve tool in the toolbar to draw the main line of the building, roughly positioning the building, and making the desired plane by push-pull tool. The graph drawing, which has the general positioning in the three-dimensional coordinate system, will be seen when the image is hidden if you describe for each layer. Then the details of the photo can be added.
- Adding texture, color and chartlet to the model, and removing useless surfaces and some auxiliary lines to simplify the model.

The modeling method of trees adopts the cross-intersection method, which is fast and has a smaller size.

Terrain is drawn using the Terrain tool embedded in Unity3D which can draw the landform according to the data collected.

Figure 3 . Flowchart Based on Image Modeling

IV. Recommendation System Interaction Design and Optimization

A. Introduction to Scenic Spots

A user can first browse all the attractions formed in photos combined with text description in the recommendation system when he/she enters it. And NGUI plug-ins can be used to realize this function. Figure 4 shows that the arrows are used for page switching and the button functions can be implemented by the Button script and Box Collider.

Figure 4 . Introduction to Scenic Spots

B. Selection of Scenic Spot Recommendation Algorithm

When the user has finished browsing, the system displays the selection page of attractions recommendation method where one can choose hot spots recommendation or personalized spots recommendation according to his/her own needs, and decide whether to fill the play time constraints. And the time parameter will be saved in the background database. Moreover, the system will operate with hot spots recommendation algorithm to give a hot route for the user if he/she chooses hot spots recommendation, while the system will jump to the user preferences page

when he/she chooses personalized spots recommendation, as shown in Figure 5.

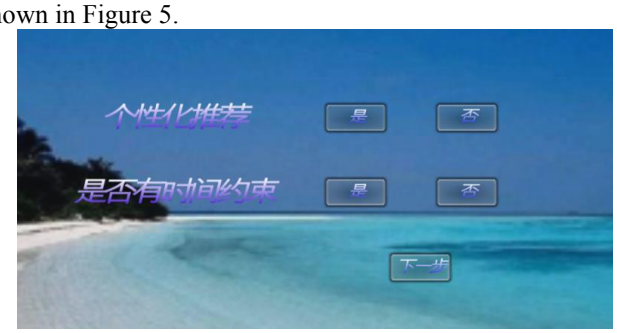

Figure 5 . Selection of Recommended Algorithm

C. Design of Character Roaming Animation

Scenic recommendation system is in the third person perspective, users can roam the various attractions according to the recommended route. At the same time you can click the right mouse button to control the movement of the protagonist. Walking, running, and idle states can switch with each other in the process of her movement, as shown in Figure 6. The switch of the protagonist's state needs Animator component added for her. It is good to produce three new states idle, walking and running for Animator Controller. One can click on any state to select the Motion in the Inspector bar to add WAIT00, WALK00_L and RUN00_F animation respectively.

Figure 6 . The Character's State Switch

D. Attractions Score

When the protagonist travels to a scenic spot of the tourist route, the NPC (Non-player-controlled character) will present the characteristics of the current attractions and provide her with travel advice, which are mainly demonstrated by text description and photos, as shown in Figure 7. And seven labels, that is, photography, shopping, food, drinks, sports, reading and history, are presented when she finished browsing. Each label has a corresponding white star for the user to evaluate. The score represents the accordance between current attraction and the label. The data will be updated in the system, so that the next recommendation will be preferable for the next user.

Figure 7 . Attractions Display

Figure 8 . Attractions Score

V. Design of Hot Spots Recommendation Algorithm

The hot spots recommendation algorithm can be transfer into the least-cost path problem of a specific node [3], and these specific nodes are the corresponding hot spots. In this algorithm, Dijkstra algorithm is needed for preprocessing in order to obtain the minimum cost and path between any two points.

A. Problem Description

In Figure 9, if you choose v_s as a starting point and v_t as the terminal point while the intermediate node needed to pass through is v_3. We can calculate four routes, including $(v_s, v_1, v_3, v_5, v_t)$, $(v_s, v_1, v_3, v_6, v_t)$, $(v_s, v_2, v_1, v_3, v_5, v_t)$, $(v_s, v_2, v_1, v_3, v_6, v_t)$ with cost of 8,16,12 and 20 respectively, stating from v_s nd going through v_3 ,then reaching v_t. Obviously, the first route has a minimum value of 8.

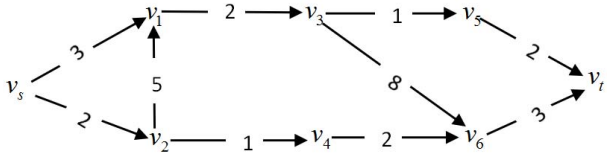

Figure 9 . A Directed Graph

B. Notations

v: a node in the directed graph g. v_s is a starting point and v_t is a destination.

V': $V' = \{v_{k1}, v_{k2}, \ldots, v_{ki}\}$, a collection of popular scenic spots.

$edge(v_i, v_j)$: an edge from v_i to v_j.

$b(v_i, v_j)$: the cost needed from v_i to v_j, including travel time and distance.

g: $g = (V, \varepsilon)$, an directed graph with set of nodes V and a set of edges $\varepsilon \subseteq V \times V$.

R: $R = \{v_s, v_k, \dots, v_t\}$, a path from the starting point v_s to the ending point v_t.

$Cost(R)$: the cost of path R.

C. Problem Definition

We calculate the minimal cost of a path from the starting point v_s, through a set of specific nodes V', to the destination v_t in the directed graph $g = (V, \varepsilon)$.

D. Algorithm Description

In Algorithm 1, Step 2 determines whether a particular node set is connected or not. Step 3 gives all permutations of specific node set V' and Step 5 gets a complete route from v_s to v_i, through V_i'. $R'(v_s, v_i)$ is calculated by Dijkstra algorithm, represents the minimum cost path from v_s to v_i, where v_i is the starting node in V_k'. $R'(v_j, v_t)$ is calculated similarly, represents the minimum cost path from v_j to v_t, which is the destination in V_k'. And $R'(V_k')$ denotes the path through V_k'. Step 6 gives the cost from v_s, through V_i' and to v_t. $Dijstra(v_s, v_i)$ is the minimal cost from v_s to v_i. $Dijstra(v_j, v_t)$ is the minimal cost from v_j to v_t and $CostMidNode(V_k')$ is the cost of a path through V_k'. Steps 7, 8 and 9 update the values of $Cost$ and R by comparison $tempMidCost$ with $Cost$.

Algorithm 1: Popular Spot Recommendation Algorithm

Input: $v_s, v_t, V'\{v_{k1}, v_{k2}, \dots\}$, g
Output: $R, Cost$
1: Initialize: $Cost \leftarrow +\infty$;
2: **if** $isConnected(V')$
3: $v_1, v_2, \dots, v_n \rightarrow FullArray(V')$
4: **for** each V_k' in $FullArray$;
5: $tempMidPath = R'(v_s, v_i) + R'(V_k') + R'(v_j, v_t)$;
6: $tempMidCost = Dijstra(v_s, v_i) + CostMidNode(V_k') + Dijstra(v_j, v_t)$
7: **if** $tempMidCost < Cost$
8: $Cost = tempMidCost$;
9: $R = tempMidPath$;
10: **end if**
11: **end for**
12: **end if**

VI. Personalized Route Recommendation Algorithm

Users' preferences become increasingly important in the process of route search and planning, see [4,5,6,7]. For instance, when a user plans a trip to a city, his/her preferences can be expressed in a series of key words, such as photos, shopping and museums, and the corresponding weights representing how much one prefers are 0.5, 0.4 and 0.1 respectively. At the same time, each attraction has multiple labels of key words and corresponding weights, which means popularity of attractions covered by the current label and the degree one prefers, such as key words for a shopping mall or a movie with weight 0.5 or 0.1, respectively. The optimal route should cover attractions marked by key words and meanwhile satisfy some constraints, such as travel time or distance, in order to meet users' demands to the most extent.

Personalized route recommendation aims to find a route that best satisfies user's preferences. And it is non-trivial to get the degree of keywords covered by the route, since simply accumulating weights of keywords cannot reflect the satisfaction of users.

Zeng proposed the ORS-KC (Optimal Route Search for Keyword Coverage)[8] for the route search. Considering the user's preferences and constraints, the coverage function is designed to calculate the number of the keywords covered by the route, with the goal of searching for a route whose value of the function is maximal under constraints.

A. Notations

K: $K = \{k_1, k_2, \dots, k_q\}$, a series of query keywords;
γ: $\gamma = \{\lambda_1, \gamma_2, \dots, \gamma_q\}$, the weights for the keywords;
Δ: constraints of time or distance;
$BS(R)$: $BS(R) = \sum_{i=1}^{m-1} b(v_i, v_{i+1})$, represents accumulation of each edge's cost in a route.

B. Keyword Coverage Function

Keywords coverage function [9] is used to compute the satisfiability of a route, which reflects the degree to which keywords covered by the route and the diversity of keywords in it. The more the types of keywords, the higher the coverage and diversity.

In the keyword coverage function, $K = \{k_1, k_2, \dots, k_q\}$ represents a set of query keywords, $\lambda_{k_q}(> 0)$ denotes the weight of keyword k_q, $R = \langle v_1, v_2, \dots, v_m \rangle$is a route from v_1 to v_m, and $KC(R)$is the keyword coverage function, as shown in Eq.(1).

$$KC(R) = \sum_{k_q \in K} \lambda_{k_q} cov_{k_q}(R) \tag{1}$$

where $cov_{k_q}(R)$ is the degree to which the keyword k_q is covered by at least one location in R, as shown in Eq.(2).

$$cov_{k_q}(R) = 1 - \prod_{v_i \in R} \left[1 - cov_{k_q}(v_i)\right] \tag{2}$$

where $cov_{k_q}(v_i)$ is the degree to which the location v_i covers the keyword k_q, as shown in Eq. (3).

$$cov_{k_q}(v_i) = \min\left[\frac{nc_{v_i}^{k_q}}{\left(1/\sum_{v_j \in V} 1\right) \times \sum_{v_j \in V} nc_{v_j}^{k_q}}, 1\right] \tag{3}$$

where $nc_{v_i}^{k_q}$ is the number of check-ins that are made at v_i labeled as k_q, $\sum_{v_j \in V} 1$is total number of visited attractions, $\sum_{v_j \in V} nc_{v_j}^{k_q}$ is the number of all check-ins labeled as k_q, and $\left(1/\sum_{v_j \in V} 1\right) \times \sum_{v_j \in V} nc_{v_j}^{k_q}$ is the average number of check-ins labeled as k_q. And the value of $cov_{k_q}(v_i)$is one if the value of $nc_{v_i}^{k_q}$ is bigger than the average number of check-ins.

C. The ORS-KC Problem

In the directed graph $g = (V, \varepsilon)$, suppose that the query sequence is $\eta = \langle v_s, v_t, K, \gamma, \Delta \rangle$, where v_s is the starting point, v_t is the target point, K is a series of key words, γ is the keyword weight, and Δ is the constraint. ORS - KC focuses on getting the route from v_s to $v_t$and satisfying the following formula.

$$\begin{cases} R = \underset{R}{\operatorname{argmax}}\, KC(R) \\ s.t. \quad BS(R) < \Delta \end{cases} \tag{4}$$

which is an NP-hard optimization problem proved by [4]. The brute-force technique guarantees an optimal solution to the ORS-KC problem, however, the exhaustive search is computationally prohibitive. To avoid enumerating all partial paths, we use a novel variant of the A* algorithm to solve this problem, which is described below.

D. Algorithm Description

ORS-KC problem can be solved by A* algorithm whose basic idea is to search candidate path set with best estimated keyword coverage value at first, and then delete the path covered with smaller keyword coverage value. It is difficult to estimate the keyword coverage function due to the fact that the objective function is submodular and the optimization problem has some constraints.

A* algorithm first establishes a search tree and then implements the breadth-first search whose order is determined by knowledge-plus-heuristic cost function. This function at the node v_n is defined as follows:

$$f^n(R_{s\to t}) = g^n(R_{s\to n}) + h^n(R_{n\to t}|R_{s\to n}) \tag{5}$$

where $g^n(R_{s\to n})$, calculated by $KC(R_{s\to n})$, is the exact keyword coverage value of $R_{s\to t}$, and $h^n(R_{n\to t}|R_{s\to n})$ is the marginal keyword coverage value of the successive path under the condition $R_{s\to n}$. $h^n(\cdot)$ and $g^n(\cdot)$ are irrelevant in the traditional A* algorithm but related in this paper. $f^n(R_{s\to t})$ will be calculated as follows:

$$f^n(R_{s\to t}) = KC(R_{s\to n}) + \frac{KC(L^{mg})}{1 - 1/\sqrt{e}} * \left(\max_{k_q \in K} \prod\nolimits_{v_i \in R_{s\to n}} \left[1 - cov_{k_q}(v_i)\right]\right) \tag{6}$$

Where L^{mg} is the solution to the greedy algorithm, see [10].

We use L_i^k to denote route label route, which stores each route and corresponding information on the node v_i, and compute as follows:

$$L_i^k = \langle R_i^k, BS(R_i^k), KC(R_i^k), f^i(R_{s\to t}) \rangle \tag{7}$$

where R_i^k represents the k-th route from v_s to v_i, stored by the maximum priority queue method.

Algorithm 2: A* Algorithm for ORS-KC

Input: $g = (V, \varepsilon)$, $\eta = \langle v_s, v_t, K, \gamma, \Delta \rangle$, $BS_{sm}(v_i, v_j)$
Output: An optimal route R
1:Initialize a max-priority $Q \leftarrow \emptyset$;
2:$R \leftarrow \emptyset$; $KC_{max} \leftarrow -\infty$;
3:Create a route label: $L_s^0 \leftarrow \langle (v_s), 0, 0, 0 \rangle$;
4:$Q.enqueue(L_s^0)$;
5:**while** Q is not empty **do**
6: $L_i^k \leftarrow Q.dequeue()$;
7: **if** $L_i^k \cdot f^n(R_{s\to t}) \le KC_{max}$ **break;**
8: Obtain R_i^k from L_i^k;
9: **for** each $edge(v_i, v_j)$ **do**
10: Create a route $R_j^l \leftarrow R_i^k \cup v_j$;
11: **if** $BS(R_j^l) > \Delta$ **continue**
12: **if** $v_j == v_t$
13: **if** $KC(R_j^l) > KC_{max}$
14: $R \leftarrow R_j^l$;
15: $KC_{max} \leftarrow KC(R_j^l)$;
16: **else**
17: Compute $f^j(R_{s\to t})$;
18: **if** $f^j(R_{s\to t}) > KC_{max}$
19: Create a route label $L_j^l = \langle R_j^l, BS(R_j^l), KC(R_j^l), f^j(R_{s\to t}) \rangle$;
20: $Q.enqueue(L_j^l)$;
21: **end if**
22: **end if**
23: **end if**
24: **end if**
25: **end for**
26: **end if**
27: **end while**
28:**if** $KC_{max} == -\infty$
29: **return** "No feasible route exists";
30:**else**
31: **return** R;
32: **end if**

In Algorithm 2, Step 1 creates an empty R, storing the current best route and KC_{max} stores the current best keyword coverage value. Step 3 and 4 create a route label containing only the starting point v_s, enqueued into Q. One

can not dequeue the candidate path labels until Q is null or all the labels in Q have an estimated keyword coverage less than KC_{max}, showed in Steps 5, 6 and 7. Step 8 takes a partial path from the dequeued labels that is to be expanded. Step 9 iterates all adjacent edges of v_i. Step 10 merges v_j, which is the neighbor of v_i, into R_i^k to form a new path R_i^k, and Step 11 skips the new path whose cost is greater than Δ. And then determine whether v_j is the target point and the keyword coverage R_j^l is greater than KC_{max} in Steps 12 and 13. One can update R and KC_{max} in steps 14 and 15 if steps 12 and13 are true. Steps 17, 18, 19 and 20 create a candidate partial path and calculate its keyword coverage estimation. And if the estimation is greater than KC_{max}, we create a new path label and enqueue it into Q. In the end, the algorithm returns an empty route where KC_{max} is never updated, or returns the optimal route R.

VII. SCENIC SPOT RECOMMENDATION SYSTEM EXPERIMENTAL RESULTS

The system operates on a Windows PC with a 4-core Intel i7-6700 3.4 GHz CPU and 16GB memory.

A. The Experimental Results of The Hot Scenic Route Recommendation Algorithm

The top five scenic spots are obtained through the questionnaire, which are XMU Times, Hibiscus Lake, Nanputuo Temple, Hibiscus Tunnel and School History Hall. One can set them as the popular attractions. If one set the number of hot attractions to be two, then he/she must visit the two scenic spots. The starting point in this paper is West School Gate while Xiamen University Seabeach is the determination. TABLE I. represents the route suggested by hot scenic recommendation algorithm and Figure 10 shows a route without time constraints via the designed system.

TABLE I. ROUTES OF HOT RECOMMENDATION ALGORITHM

Number	Route	Hot Spots	Cost (min)
1	West School Gate, Xiamen Time, School History Hall, Chen Jiageng Statue, Lu Xun Square, Human Museum, Xiamen University Seabeach.	Xiamen Time	94
2	West School Gate, Xiamen Time, School History Hall, Chen Jiageng Statue,Hibiscus Lake, Lu Xun Square, Human Museum, Xiamen University Seabeach.	Xiamen Time, Hibiscus Lake	103
3	West School Gate, Nanpu Temple, South School Gate, Library, Chen Jiageng Statue, School History Hall, Xiamen Time, Lu Xun Square, Human Museum, Xiamen University Seabeach	Xiamen Time, Hibiscus Lake, Nanputuo Temple	189

Figure 10 . Route Map

B. Experimental Results of Personalized Route Recommendation Algorithm

The recommendation system will present specific route based on the consuming time and users' preferences inputted by users, as shown in Figure 11. The route recommended for the user in Figure 11(a) differs from that in Figure 11with the same input time since their preferences are different. Concretely, the input time is 3 hours, and labels in Figure 11 are shopping, food, drinks and taking photos with weights of 0.44, 0.22, 0.22 and 0.12 respectively while labels in Figure 11(b) are history, reading, taking photos and sporting with weight of 0.4, 0.4, 0.1 and 0.1 respectively. Hence, the route in Figure 11(a) is West School Gate, Nanputuo Temple, South School Gate, International Exchange Center, Keli Supermarket, Qinye Restaurant, Kat Home Department Store, 1921 Cafe, Luyinhai Supermarket and Xiamen University Seabeach, including Keli Supermarket, Kat Home Department Store and Luyinhai Supermarket for shopping label, Qinye Restaurant for food label and 1921 Cafe for drinking label. And the route in Figure 11(b) is West School Gate, Xiamen Time, School History Hall, Chen Jiageng Statue, Library, South School Gate, International Exchange Center, Keli Supermarket, Qinye Restaurant, Human Museum and Xiamen University Seabeach. These results shows that the routes run by our system cover users' preferences.

With the ongoing functioning of recommendation system, users' evaluation of the attractions saved on the background will have an impact on the system, constantly updating the data. That displays in Figure 12 with the same input time and personal preferences. The route initially is West School Gate, Xiamen Time, School History Hall, Chen Jiageng Statue, International Exchange Center, Keli Supermarket, Qinye Restaurant, Kat Home Department Store, Human Museum and Xiamen University Seabeach, while it is West School Gate, Nanputuo Temple, South School Gate, International Exchange Center, Keli Supermarket, Hibiscus Lake, Luyinhai Supermarket and Xiamen University Seabeach after the system runs for a period of time, as shown in (a) and (b) of Figure 12 respectively.

(a)

(b)

Figure 11 . The Influence of Individual Preferences on the Algorithm

(a)

(b)

Figure 12 . The Impact of Users' Evaluation on the Algorithm

VIII. CONCLUSION

This work mainly designs the spots recommendation system based on Unity3D. We use the virtual reality game engine Unity3D to realize the roaming of the scenic spots. In the meanwhile, we use two route recommendation algorithms to realize the function, and experimental results shows that the system generates a route with various scenic spots and meets the needs of users. Moreover, scoring for attractions is implemented for constantly updating system data so as to make the recommended results as user-friendly as possible.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China (Grants 61375070, 61562033 and 11671335).

REFERENCES

[1] B. M. Oh, F. Durand, and M. Chen, "Image-based modeling and photo editing", Proceedings of the 28th annual conference on Computer graphics and interactive techniques ACM, 2001, pp. 433-442.

[2] H. Han, "Research on 3D Modeling Based On Hand Sketching", Nanjing university of aeronautics and astronautics, 2009.

[3] S. Huang, D. Hu, and Y. Jiang, "The Shortest Path Algorithm of The Selected Intermediate Node Set", Computer engineering and application, 2015, 51 (11).

[4] X. Cao, L. Chen, G. Cong, and X. Xiao, "Keyword-aware optimal route search", Proceedings of the Vldb Endowment, 5(11), 2012, pp. 1136-1147.

[5] F. Li, D. Cheng, M. Hadjieleftheriou, G. Kollios, and S. Teng, "On trip planning queries in spatial databases", Journal of Combinatorial Optimization, 31(1), 2005, pp. 1-16.

[6] M. Sharifzadeh, M. R. Kolahdouzan, and C. Shahabi, "The optimal sequenced route query", VLDB Journal, 17(4), 2008, pp. 765-787.

[7] C. Chekuri and M. Pal, "A recursive greedy algorithm for walks in directed graphs", Foundations of Computer Science, 2005. FOCS 2005. IEEE Symposium on. IEEE, 2005, pp. 245-253.

[8] Y. Zeng, X. Chen, X. Cao, S. Qin, M. Cavazza, and Y. Xiang, "Optimal route search with the coverage of users' preferences", International Joint Conference on Artificial Intelligence, AAAI press, 2015, pp. 2118-2124.

[9] K. El-Arini, G. Veda, D. Shahaf, and C. Guestrin, "Turning down the noise in the blogosphere", ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2009, pp. 289-298.

[10] S. Khuller, A. Moss, and J. Naor, "The budgeted maximum coverage problem", Information Processing Letters, 70(1), 1999, pp. 39-45.