Journal of Information Hiding and Multimedia Signal Processing ©2017 ISSN 2073-4212
Ubiquitous International Volume **8**, Number **3**, May **2017**

Efficient Path Finding Method Based Evaluation Function in Large Scene Online Games and Its Application

Xingquan Cai[1], Shiyu Li[1], Xin Wu[1], Hao Lu[1,2]

[1]School of Computer Science, North China University of Technology
Beijing 100144, China
xingquancai@126.com*
[2]North China Institute of Computing Technology,
Beijing 100083, China

Received July, 2016; revised March, 2017

Abstract. *In large scene online games, efficient path finding is an important technology. In this paper, we provide an efficient path finding method based evaluation function in large scene online games. Firstly, we divide the big scene map into many map blocks, build map areas, organize them in index table. After that, we load the scene map dynamic. Then, we set the evaluation function, and we use the heuristic search to complete the path finding in one map area or cross map areas. We smooth the path and roam in large scene online game. Finally, we implement our system and provide the results. The experiment results show that the effect of dividing the scene map into map blocks of 256*256 is the best, when dynamic load scene map. And our methods is better than the A* path finding method, especially in finding smooth and natural path when virtual character find path cross map areas. Our system has been used in practical projects, and our system is running smoothly and stably.*

Keywords: Large scene online games; Path finding; Evaluation function; Map blocks; Cross map areas

1. **Introduction.** With the rapid development of virtual reality, computer graphics, artificial intelligence and interactive media, online games develop quickly, especially in massive multiplayer online games. Generally, the world map of multiplayer online games is very large. So, there are several problems while playing, such as delay in loading scene data, stuck in switching game maps, low efficiency in virtual character finding path automatically. All these problems make the games experience very poor. Therefore, it is necessary to research a path finding method in large scene online games to improve the efficiency of the big scene map loading and virtual character finding path automatically.

2. **Related works.** The path finding of virtual character is an essential part of artificial intelligence development in a game. Its main purpose is finding a shortest and minimum cost path according to the terrain and obstacles of different scene maps. However, the map of the large scene multiplayer online game is so bigger. So it is necessary to research the game scene map loading and organization to improve the efficiency of path finding. Game scene map loading is an important part of a game, because the speed of loading directly affects the experience of players. There have been various methods of scene map loading. In 2009, Liu [1] used static loading and dynamic preloading to realize the continuously fast

display of large scale geographic information when used satellite images to stitch big digital scene maps. In 2010, Yuan [2] proposed a dynamic on-demand loading method based on partitioning the graphic elements and building block index to realize fast querying, loading and displaying the map data of GIS. Wang [3] used Bing Maps section to get the scene data and produced the web map service. In 2011, Li [4] presented a 3D multi-core parallel loading system using for rendering scene. Loading the scene with OGRE, multi-thread creation and synchronization using OpenMP and setting the number of threads dynamically is adopted in parallel programs in the multi-core computers. In 2014, Lu [5] implemented a service-oriented space situational information system according to the concept of Service Oriented Architecture. The key techniques involved in the system, such as the design of the service, data management and distribution, display and switch of the scene, adding and display of the target, loading and saving of the scene.

As to the path finding, in 2008, in order to realize the motion simulation of vehicle in the virtual airport, Yuan [6] organized the global map by the points and lines of space. He organized the local map by the more detailed grid representation, and used heuristic search to complete path finding of cars. In 2010, Liang [7] presented a path finding method for massive multiplayer online game based on locating points and reusing paths. This method can reduce unnecessary space search and reduce the load of server. In 2011, Meng [8] used RSG model to organize the 3D scene terrain data. It can generate an editable and more detailed navigation mesh data and complete path finding globally. And Zhang [9] proposed a high available distributed navigation method based on access vestige fusion and analyzed in a series of simulation. In 2012, Zhang [10] proposed a path planning based on heuristic algorithm and improved valuation function model. In 2013, Chen [11] improved BP neural network and designed automatic path-finding. In 2014, Liu [12] proposed a MOV A* algorithm based on price vector multi-objective A* algorithm and realized the multi characters intelligent path planning in a game. And Yu [13] presented a fast and high-efficient dynamic path finding method. It reflected the interaction between the actual units in a game [14]. In 2016, Zhang [15] proposed a novel disaster-tolerant navigation algorithm based on electric power equipment coordinate vestiges, in order to achieve the electronic map disaster recovery and automatic routing. He presented the algorithm core idea, unit structures, vestige data structures and processing flows. The information fusion methods and electric power equipment coordinate vestiges are used to generate feasible ways automatically. And the algorithm used bi-dimensional space to match way information and way-finding needs.

So in this paper, we mainly research a path finding method in large scene online games. We organize scene maps effectively, load scene maps dynamically and improve the efficiency of path finding.

3. Efficient Path Finding Method Based Evaluation Function.

Usually, in large scene online games, the static scene map loading method costs such a long time and the game experience is so poor. Therefore, it is necessary to load the scene map dynamically and then complete path finding.

3.1. Dynamically Loading Large Scene Map.

In a large scene online game, the game scene map loading is an important foundation. The loading efficiency affects the real game experience. Usually, the small scene game used the static loading scene. In this paper, we use a dynamic loading scene map method to improve the speed of loading. Firstly, we divide the large scene map into many square map blocks. Then, we form the map blocks into many map areas according to the screen resolution, and we also build index table for each map area. Finally, we load map areas dynamic.

3.1.1. *Dividing the Big Scene Map into Map Blocks:* Usually, there are three methods of map segmentation often used in the big scene games, including regular hexagon segmentation method, NPC (Non Player Character) segmentation method and square segmentation method. The hexagon method is that divide the scene map into many hexagonal map blocks in same size. The NPC method is that determine the map blocks based on the coordinate of NPC. The square method is that divide the scene map into same square map blocks, then build the index of them and do various algorithm organizations. The square segmentation method is more efficiency and easy to implement, so we use this method to divide the big scene map.

3.1.2. *Building Map Areas and Index Table:* It needs to build map areas in different sizes according to resolution of different displays, because the resolution of different players may be different. As Figure 1 shows, each big rectangle is a map area, and each small rectangle is a map block of 256*256. In order to adapt to the display resolution of 1280*768, each map area consists of 15 map blocks using 5*3 arrangements so that a map size screen displays is just a map area size.

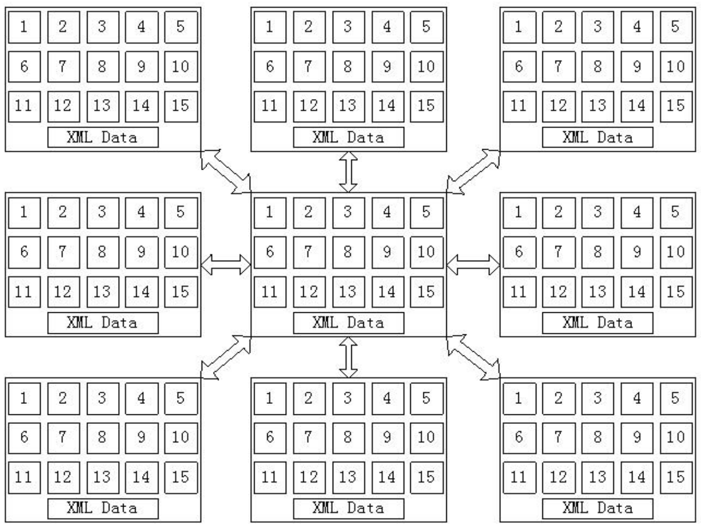


FIGURE 1. Building map areas and index table adapting to the display resolution of 1280*768

Usually, we should avoid appearing white screen issue when virtual character moves to the map border. So we should continually load and real-time render the map area which virtual character will arrive. We use a list to cascade every map area which records the index information about around map areas. When need to load a new map area, loaded by reading the adjacent map areas index information from the configuration information of loaded map areas. As Figure 1 shows, each map area records the relative position of containing map blocks, the index values of map area and surrounding 8 map areas in an XML file.

3.1.3. *Dynamically Loading the Map Area:* In the process of playing game, player may click on the adjacent map area, and player may also click on the map area far from the current map area. Therefore, the dynamic loading map area divides into adjacent map area loading and cross map area loading.

(1)Adjacent map area loading

We compare the relative distance of virtual character moves at a time with the width and height of display area, in order to determine the map area dynamic loading time

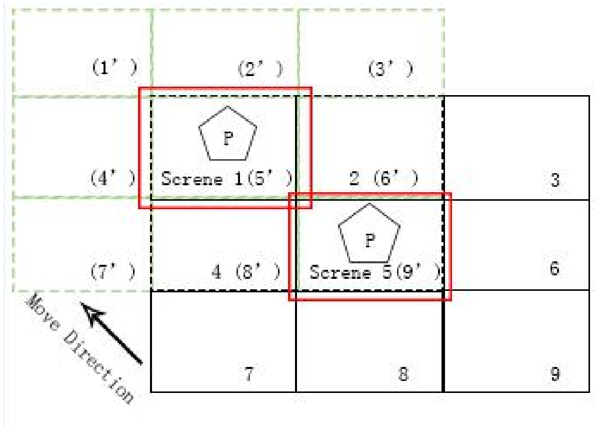


Figure 2. Loading map areas while virtual character moving

and which direction map area needs to be loaded. When the virtual character moves, relative horizontal displacement is greater than half width of the display area, or relative vertical displacement is greater than half height of the display area, or these two cases both appear, determine the virtual character has moved to the adjacent map area and need to load a new map area. As shown in Figure 2, the relative horizontal displacement is negative and the absolute value is greater than SceneWidth/2 and the relative vertical displacement is negative and the absolute value is greater than SceneHeight/2, system determines the virtual character move from map area 5 to map area 1. The map area 1', 2', 3', 4', 7' will be loaded and the map area 3, 6, 7, 8, 9 will be removed.

(2)Cross map area loading

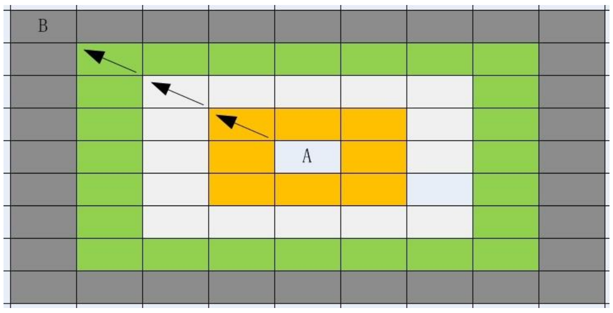


Figure 3. Process of cross map area loading

In general, the application will provide a thumbnail of the big scene in a large scene game. Player can click the place name in the thumbnail to let the virtual character move to a location far from the current map area. When player clicks a far place in the thumbnail, the application will set the index value of the selected map area as target and search outward layer by layer from the current map area and record the relevance between each node of the current layer and node of last layer. After finding the target map area, backtrack search path and save the index values which the path through in a loading sequence table. It can directly load the target location map area and the surrounding 8 map areas. It can also load the map area in sequence according to the index of loading sequence table. As Figure 3 shows, area A is the current map area. When player click area B, the application search map areas layer by layer until find the target map area B, determine the map areas loading order and complete the virtual character roaming to area B from area A.

3.2. **Finding Path Based on Evaluation Function.** When using interactive tools, it needs to find a path from starting point to target point. However, there are passable areas, mask areas and impassable areas in one game scene map. So path finding becomes not easy to complete. This paper adopts a path finding method based on the evaluation function. This method will effectively find path in current map area or across map areas in a large scene game.

3.2.1. *Finding Path in Current Map Area:* When the player selects the target location in the current map area, we use heuristic search based on evaluation function, to find the best path from starting point to target point. Evaluation function is shown in formula (1). Among them, G(x, y) is the actual cost of searching from the starting point to the current search point. H(x, y) is the estimation cost of searching from the current search point to the target point, namely the sum of horizontal distance and vertical distance between two points.

$$F(x, y) = G(x, y) + H(x, y) \tag{1}$$

The search process is as follows:

Step1: Calculate the evaluation function value F(x, y) of passable points around the starting point.

Step2: Select a point from passable points which value of F(x, y) is the minimum. Save this point in the path table and calculate the F(x, y) of passable points around this point which have not calculated.

Step3: Repeat Step2 until appear a point its H(x, y) is 0. It is the target point. Save it in the path table.

Step4: The path table finally obtain is the path from the starting point to the target point.

3.2.2. *Finding Path across Map Areas:* In a large scene game, when the virtual character moves across map areas, the target point located map area has not been loaded into the application, so the system can not calculate the evaluation function in the process of searching path. The method of finding path across map areas calculates the evaluation function according to the loaded map areas and then finds a satisfactory path. The process is shown in Figure 4.

Step1: According to the index value of map area where the target point is and the level traversal sequence, determine the map areas loading sequence from area of the starting point to area of the target point.

Step2: According to the map areas loading sequence, determine the target point of current map area. If the adjacent map areas intersect at one point, the target point of current map area is the intersection point. As shown in Figure 5, point D1, D2 and D3. If the adjacent map areas intersect at on edge, the target point of current map area is the midpoint of the edge. As shown in Figure 5, point D4.

Step3: Based on the method of path finding in current map area, search the optimal path from the starting point to the target point of current map area. Then set the current target point as the starting point of next search.

Step4: Repeat Step2 and Step3, until find the ultimate target point. The path table is the ultimate path from the starting point to the target point.

Step5: Smooth the path. Because the obstacles are dense and the spaces of obstacles are irregular, the path of virtual character through obstacles in the scene is excessive zigzag. So it needs to smooth the path to improve the game experience. The node saved in the path table is coordinate of path point. There are three cases of slope of the adjacent

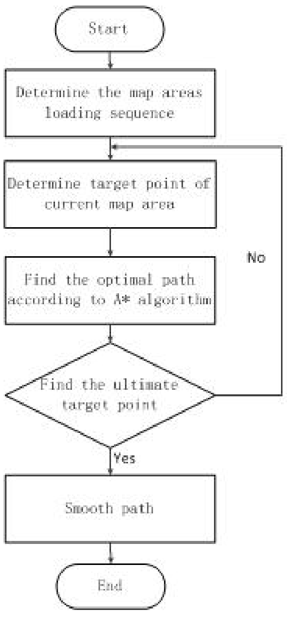

Start
Determine the map areas loading sequence
Determine target point of current map area
Find the optimal path according to A* algorithm
Find the ultimate target point
No
Yes
Smooth path
End

Figure 4. Process of path finding across map areas

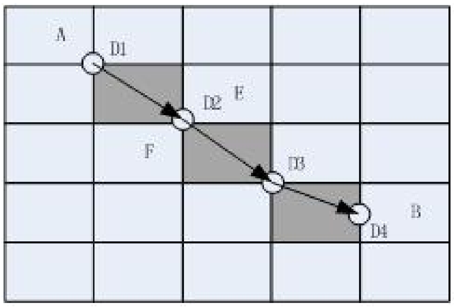


Figure 5. Path finding across map areas

points: 1, -1, and 0 (infinity is calculated as 0). We use the slope changes to judge the curving section. Firstly, calculate the slope of the adjacent points and store in the smooth path table. Then, find path section that the product of adjacent elements in the smooth path table is -1. If the products of three pairs adjacent elements are -1, we determine the path section is curving. Finally, merge the curving path sections according to the direction of real curving path section. So, it can get a smooth path from the starting point to the target point across map areas.

4. **Results.** In order to verify our path finding algorithm in large scene online games, we have implemented the dynamic loading large scene map method and a path finding method based on evaluation function. We design and realize a large scene online virtual university game system using our algorithm well. The hardware devices of our system include Intel Core i5-3317U 1.7GHz and NVIDIA GeForce740. The software environment is Window7, Visual Studio2008, Flash Builder4.6 and SQLServer2008.

4.1. **Dynamic Loading Big Scene Map Experiments.** In order to verify our method, we present a dynamic loading large scene map experiment. We provide a map editor to place the barriers, just as Figure 6 shows. We use a large scene map that resolution is 5120*5120. Set the resolution of game scene as 1024*1024 and the size of map area is

Table 1. Time of loading map areas

Size of map block	Before compression	After compression	Block number in a area	Time of loading the current map area	Time of loading the squared figure areas
1024*1024	3MB	100KB	1	20(ms)	250(ms)
512*512	0.75MB	60KB	4	80(ms)	1160(ms)
256*256	192KB	25KB	16	480(ms)	4460(ms)
128*128	48KB	10KB	64	2120(ms)	19120(ms)
64*64	12KB	5KB	256	8880(ms)	763600(ms)

1024*1024. So there are 25 map areas in the scene. Adjust the size of map blocks and load map areas in different size. After 100 experiments, count the time of loading the current map area and the squared figure areas then calculate the average. Experimental result is shown in Table 1. Map is compressed by JPG format. From the results, the size of map blocks is smaller, the time of loading the current map area and the squared figure areas is longer. However, the actual net speed of general players is about 200KB. Therefore, the size of map blocks is more bigger, the efficiency of server transmission is more lower. When the size of map blocks is 256*256, the efficiency of dynamic loading is higher, just as Figure 7 and Figure 8 show .

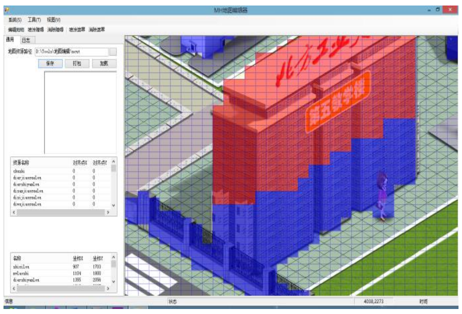


Figure 6. Using map editor to place the barriers in the large scene

Figure 7. Loading the scene by traditional static loading method

4.2. **Virtual Character Path Finding Experiments.** We have realized the virtual character path finding method based on evaluation function in games and compared with the traditional A* path finding method. Using traditional A* method to find path in map area is more effective but the path is zigzag especially when there are too many obstacles

FIGURE 8. Loading the scene by our dynamic loading method

FIGURE 9. Virtual player through the cross area to reach the destination

in the scene. Our method can better complete path finding across map areas and smooth the path. Path found by A* method is shown in Figure 10. Path found by our method is shown in Figure 11. Obviously, path found by our method is smooth and natural.

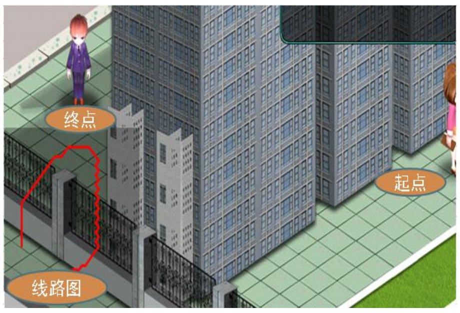


FIGURE 10. Path finding by traditional A* method

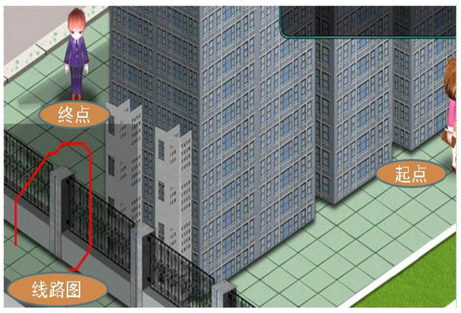


FIGURE 11. Path finding by our method

4.3. **Large Scene Online Virtual University Game System.** Large scene online virtual university game system is designed about the campus life of North China University of Technology. This system uses the form of multiplayer online game and the method of completing tasks to help new students to know the process of register, military training, extracurricular activities, final exams and final graduation in short time. The client system is divided into four main modules, namely, task management, character roaming, campus display and character communication. The task management module includes masterstroke tasks and slide tasks. The system is organized by task orientation. The character roaming module includes campus map navigation and real-time interactive roaming in two ways. Campus map navigation provides the bird's eye map of the campus and new students can quickly arrive at any building by the map. Real-time interactive roaming can let new students go to any corner by interacting. Character communication module is used to communicate between students.

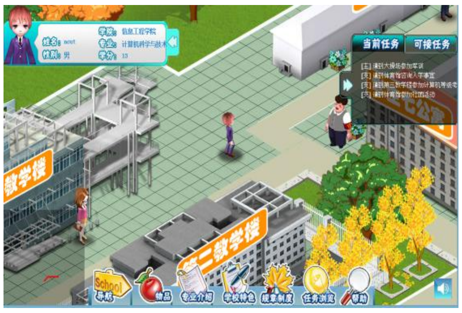
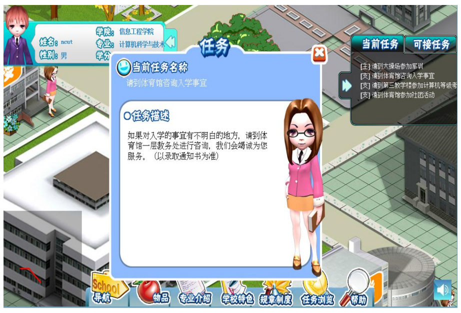


FIGURE 12. One screenshot of game scene



当前任务名称
请到体育馆咨询入学事宜
任务描述
如果对入学的事宜有不明白的地方，请到体育馆一层教务处进行咨询，我们会竭诚为您服务。（以录取通知书为准）

FIGURE 13. Description of a game task

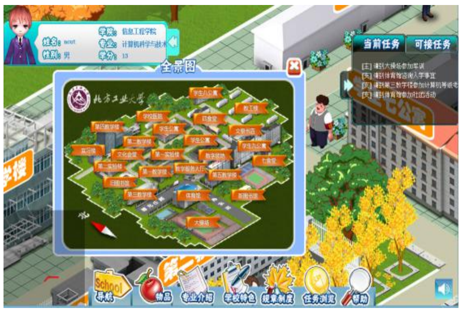


FIGURE 14. Game panorama

The game use the method proposed by this paper well. Map initialization is quick. Map transition is smooth. The path that virtual character finds is reasonable. Figure 12 is a screenshot of game scene. Figure 13 is the description of a game task. Figure 14 is the game panorama. Click the area name in the map, send quest to map server according to the index of map area. Dynamically load the scene map to the client and real-time rendering in the process of roaming and complete path finding in the same time.

5. **Conclusions.** In this paper, we provide one efficient path finding method based evaluation function in large scene online games. Firstly, we divide the big scene map into many map blocks, build map areas, organize them in index table. After that, we dynamic load the scene map. Then, we set the evaluation function and use the heuristic search to complete the path finding in one map area or cross map areas. We smooth the path and roam in large scene online game. Finally, we implement our system and provide the results. The experiment results show that the effect of dividing the scene map into map blocks of 256*256 is the best, when dynamic load scene map. And our methods is better than the A* path finding method, especially in finding smooth and natural path when virtual character find path cross map areas. Our system has been used in practical projects, and our system is running smoothly and stably.

Further work includes optimizing our method to judge whether virtual character moves to the loaded map areas and avoid repeated loading.

Acknowledgment. Supported by the Funding Project of National Natural Science Foundation of China (No. 61503005), the Funding Project of Natural Science Foundation of Beijing (No. 4162022), the Funding Project of National Science and Technology Support Plan (2012BAF84F02), and the Funding Project of Great Wall Scholar Reserve Training Program in North China University of Technology (No. NCUT2015006).

REFERENCES

[1] X. M. Liu, L. L. Lin, S. D, Jiang, Loading of the Geographic Pictures for Digital Map , *Computer Applications*, vol. 28 , no. 10 , pp. 34-41, 2009.

[2] M. Yuan, L. Y. Zhang, Application of Dynamic On-demand Loading Algorithm in GIS Map , *Computer Engineering*, vol. 6, no. 15, pp. 245-247, 2010.

[3] X. D. Wang, H. P. Liu, Y. Qiao, The Utilization of Bing Maps Tiles Data to Implement Web Map Service , *Remote Sensing For Land & Resources*, vol. 22 , no. 2 , pp. 122-127, 2010.

[4] Z. Li, X. W. Zheng, Application of Multi-core Parallel Technology in 3D Scene Loading , *Computer Engineering*, vol. 37 , no. 6 , pp. 245-247. , 2011

[5] W.J. Lu, Q. Xu, C. Z. Lan, Q. S. Shi, at tel., Design and Implement of Service-Oriented Space Situational Information System , *Journal of System Simulation*, vol. 26 , no. 10 , pp. 2452-2457, 2014.

[6] J. B. Yuan, P. L. Mu, at el., Efficient path-finding algorithm of virtual vehicle in large-scale scene , *Computer Engineering and Design*, vol. 29 , no. 10 , pp. 2622-2625, 2008.

[7] Y. Liang, G. Zhou, Path finding algorithm in massive multiplayer online games based on anchor points and paths reuse , *Journal of Computer Applications*, vol. 30 , no. 12 , pp. 3215-3217, 2010.

[8] Y. Meng, B. Q. Liu, Research on Path-finding Algorithm of Player in 3D Scene , *Journal of Wuhan University of Technology*, vol. 33 , no. 12 , pp. 125-130, 2011.

[9] J. L. Zhang, Z.Lin, A High Available Distributed Navigation Alogorithm Based on Access Vestige Fusion , *Journal of Sichuan University*, vol. 43 , no. 3 , pp. 119-122, 2011.

[10] B. Q. Zhang, Path Planning Based on Heuristic Algorithm , *Computer Simulation*, vol. 29 , no. 10 , pp. 341-343, 2012.

[11] T. T. Chen, K. Li, X. R. Du, Design and implementation of automatic path-finding based on improved BP neural network , *Computer Engineering and Design*, vol. 34 , no. 11 , pp. 3989-3995, 2013.

[12] D. R. Liu, Q. Chen, T. Lin, Path finding using new Multi-objective A* for video game NPC , *Application Research of Computers*, vol. 31 , no. 7 , pp. 91-95, 2014.

[13] S. Yu, at tel., Interactive Path-planning Method Based on Artificial Potential Field in Game Scenarios , *Computer Science*, vol. 41 , no. 2 , pp. 131-135, 2014.

[14] T. Z. Qiao, X. P. Guo, at el., Cooperative Task Assignment Simulation of Multi-UAVs in Dynamic Environments , *Journal of System Simulation*, vol. 28 , no. 9 , pp. 2126-2132, 2016.

[15] J. L. Zhang, Novel Disaster-Tolerant Navigation Algorithm Based Electric Power Equipment Coordinate Vestiges , *Journal of University of Electronic Science and Technology of China*, vol. 45 , no. 2 , pp. 276-281. , 2016