Researching on AI Path-finding Algorithm in the Game Development

Xiangguang He
Department of Information Engineering
Binzhou Polytechnic
Binzhou, China
e-mail: hexiangguang@bzpt.edu.cn
pb_hxg@126.com

Yaya Wang
Department of Information Engineering
Binzhou Polytechnic
Binzhou, China
e-mail:wangyaya@bzpt.edu.cn

Yanyan Cao
Department of electrical engineering
Binzhou Polytechnic
Binzhou, China
e-mail: yaya_sd@163.com

***Abstract*—Path-finding is a basic topic which AI in real-time game. This paper for the characteristics of path-finding in the games and based on analysis and research the classical AI path-finding algorithm introduces a research on two-tiered algorithm based on Uniform block which to improve quality that searching shortest path in two points. At last, the experimental results illustrate the effectiveness of the proposed algorithm.**

Keywords-Artificial intelligence ; Path-finding ; A* algorithm; two-tiered path-finding algorithm; Uniform block

I. INTRODUCTION

By the development of the electronic game industry, more and more games using artificial intelligence to improving their interesting. Path-finding is used in the process of most game development. Finding the Shortest path which is the roles from the source to reach the target point in the map quickly and accurately is the basic and important part of the Game AI. Path-finding is the basic ability of no-player character[1]. At the meantime, by the complexity of game scene path-finding is a challenging topic for game developer in programming.

II. CLASSICAL AI PATH-FINDING ALGORITHM

There are two different methods in game path-finding, heuristic search and non-heuristic search. Non-heuristic search is a null information guide. Only the system executes a call in random rules in turn. This search strategy efficiency is low and takes too much space and time calculation; Heuristic search is with some heuristic information in particulars; Choose the accessible node as the next expanded node,Dynamically calls the appropriate rules to find the optimum relation.

A. Non-heuristic Dijkstra Algorithm[1]

Dijkstra algorithm is a classic in shortest path algorithm, for calculating the shortest path which one node to all other nodes. This algorithm set up a domain for each node to recording the shortest path from the beginning node to them. Dijkstra algorithm can get the optimal solution of shortest path. However, it travels a lot of nodes and low efficiency.

B. Heuristic A Algorithm[3]*

A* algorithm is a typical heuristic search. By using the evaluation function to evaluate the value of decision-making, we can choose the first strategy and achieve the prioritization of breadth-first search. We can denote the evaluation function as $f(n)=g(n)+h(n)$, $f(n)$ shows the evaluation function which node n from the initial point to goal. $g(n)$'s value is certain. It shows the real cost which from the initial point to goal in abstract space. $h(n)$ shows the evaluation function which n from the initial point to goal. It is heuristic information which is according to the real problem. $h(n)$ is the conditions to ensure use of A * to find the shortest path algorithm to find. If $h(n)\leq$ the real distance from n to goal, we can find the optimum solution but the search range large, low efficiency; If $h(n)$>the real distance from n to goal we can not get the optimum solution, but the search range of small, high efficiency.

III. TWO-TIERED PATH-FINDING ALGORITHM BASED ON UNIFORM BLOCK[4]

There are lots of nodes in game scenes in practical applications. It is difficult to find out the path between two specified nodes in very short time simplely use Dijkstra algorithm,A* algorithm or other theoretical algorithms. We can use Hierarchical path-finding. It processes a group of nodes of the original map as a macro-node(named Cluster).In path--finding, it first finds out the path in the map of Cluster and then in the internal of the Cluster. Combine these paths together to make up the final path. This is two-tiered path-finding.

C. The Steps of Algorithm

From the starting node, make a number of nodes horizontal as the width of the Cluster, a number of nodes

978-1-4673-2467-0/12/$31.00 ©2012 IEEE

longitudinal as the height. Cluster is taken as a kind of index not a path-finding node.

1) Analyse the Uniform block of every Cluster.

2) Traverse Uniform blocks between Clusters in the order of Cluster,compute the entrance.

3) Import the start node and object node for path-finding.

4) Search the block the start point and target point belong to. Use A algorithm in uniform blocke for node units for abstract path-finding.*

5) Lisi Obtain the entrances among block in abstract path.

6) Compute the detailed path among entrances.

7) Compute the detailed path from the start point to the start entrance and the detailed path from the last entrance to the target point.

8) Combine the paths.

D. Detailed Algorithm Procedure

1) Uniform block is a continuous area without any obstacle.But such an area is very large in the whole map. So it is restricted in the internal of a cluster as in Fig.1.

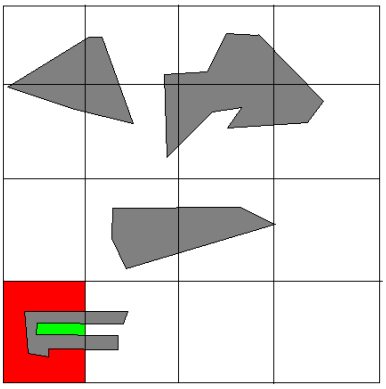

Figure 1

In the cluster at the left bottom of Fig.1, gray indicates obstacles, red and green show the Uniform block. We can see the cluster includes two Uniform blocks without any connection.

Analyze the Uniform block of every Cluster is one of the most important and also the most complex part logically. The simple description is to connect all the non-obstacle rows by row to form a Uniform block. The complex is that it needs patience for code realization. The procedures are described in Fig.2.

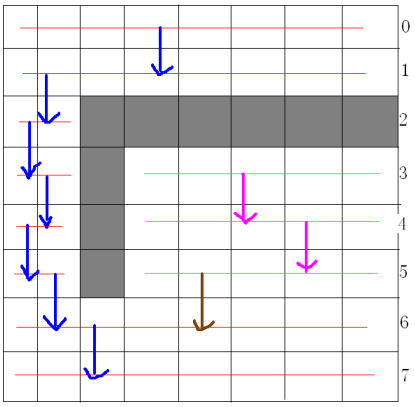

Figure 2

a) Traverse the nodes in a row. This row will be the beginning of a Uniform block if there are no obstacles. Continue traversing if you encounter obstacles until finding non-obstacle node, another Uniform block is created. Suppose in row zero there are on obstacles, this row belongs to one uniform block. In the third row, the first two boxes and the last five are seperated, then, these two parts belong to two Uniform blocks.

b) Continue traversing the next row. Estimate if there are connections between the analyzed one or several sections in this row and the last row Add this section to the Uniform block if there are connections.With situations shown in the fifth and sixth rows, combine the red and green areas together, the originally two blocks become a Uniform block.

c) Repeat procedure b until a cluster is traversed.

2) Find all the clusters and analyse the connectivity among them, which is called inter-edge.

A inter-edge is a node connecting the two Uniform blocks, this node is called entrance. There may be more than one inter-edge between two Clusters but we save only one entrance here. As two Clusters in Fig.3, the green Uniform block and the yellow can generate entrance not only in position 1 but also in 2. This algorithm chooses the one nearest to the centre as the final entrance and abandon the other.

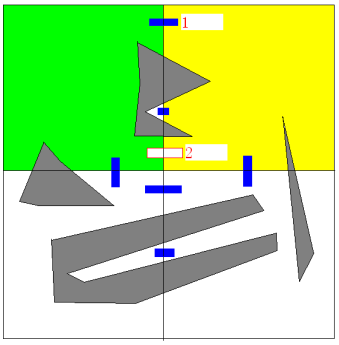

Figure 3

3) The above steps are preprocessing and be saved in a file. The next time we use this map for path-finding, we can directly read it from the file, including the obstacle information, Cluster distribution information, the entrance information between block and so on. Then, transfer to the start point and the target point for path-finding.

4) Obtain the Cluster that the node belongs according to the node's coordidate. Traverse all the uniforme block in the Cluser to decide which blocke the node belongs to. Obtain the uniform blocke that the start point and target point belongs to, then we can use A algorithm in uniform blocke for node units for abstract path-finding.*

5) Compare two adjacent blocks in the abstract path to get the entranc of them. We can get these from the preprocessing information.

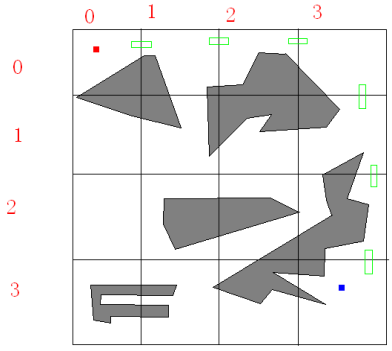

Figure 4

As in figure 4, suppose the red dot is the start point and the blue as the target point. The obtained abstract path has been connected by the green in the entrances.

*6) We can use one level A*algorithms to compute the detailed path among entrances. As two adjacent entrances must be in the same blocke, the scope of the A*algorithms path-finding will be very samll. The expenditure will be small and it will find the path easily. The path figure will become that shown in Fig.5.*

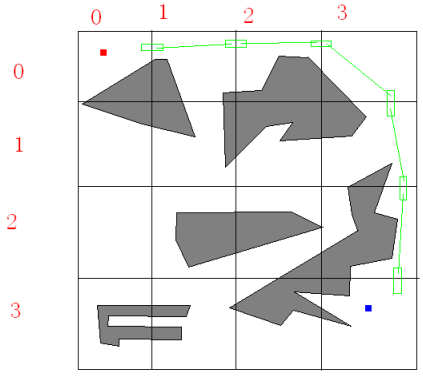

Figure 5

*7) Similarly,use one level A*algorithms to compute the path which from the start point to the last entrance and the path which from the last entrance to the target point.*

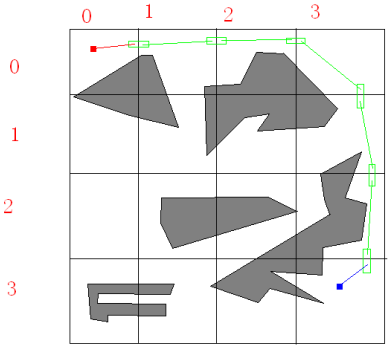

Figure 6

8) The path generated finally is the final path as shown in figure .6.

It can be seen that the algorithm is more efficient, so the player will not notice the existence of finding time.

IV. CONCLUSION

This paper analyzes process of research on two-tiered algorithm based on Uniform block which abstracted the node completely. Not consider the shape of the node, but only consider connectivity, so it has a flexibility and high efficiency. However,only one entrance be saved between two unform blocks, loss of accuracy and there are shortcomings in optimization of the path.

REFERENCES

[1] B.Stout,Smart Moves. Intelligent Path-finding. Game Developer Magazine,1996

[2] Y.Weimin,W.Weimin. Data Structure. Tsinghua University Press

[3] Patrick Lester. A* Path-finding for Beginners. http://www.gamedev.net/reference/articles/article2003.asp

[4] Nathan Sturtevant,Michael Buro. Partial Path-finding Using Map Abstraction and Refinement.