Journal of Information Technology and Engineering
Vol. 2 No. 1 (June, 2017)

Simulating Crowd Evacuation with a Leader-Follower Model

Qingge JI[1,2,3] & Can GAO[1,4]

[1]School of Information Science and Technology, Sun Yat-sen University, Guangzhou, 510275, P.R.C.
E-mail: issjqg@mail.sysu.edu.cn

[2]Key Laboratory of Digital Life (Sun Yat-sen University), Ministry of Education, P.R.C.

[3]State Lab of CAD & CG, Zhejiang University, Hangzhou, 310027, P.R.C.

[4]Information Networking Institute, CMU, USA.

Received: 13th September 2016 Revised: 24th October 2016 Accepted: 05th December 2016

Abstract: *Emergency situations involving a large number of people may develop into disasters. One example of such emergency situation is fire evacuation in a building or crowded area. Although crowd evacuation simulation has been an active research area for many years, little research focuses on simulating the evacuation process with a* Leader-Follower *Model which shows the role of leaders in the evacuation. In this paper, we have simulated crowd evacuation process from a dancing hall. A crowd includes several groups with each group having a leader and some followers guided by the leader. Leaders are responsible for finding the evacuation path for their followers. The objective of this simulation is to show the effect of different number of leaders on the efficiency of evacuation. A dynamic grouping algorithm based on A* Algorithm has been proposed to simulate the dynamic grouping phenomena with a number of leaders. A prototype system has also been developed to verify the effectiveness of the proposed algorithm.*

Keywords: *Virtual Crowd, Leader, Cellular Automata, Multi-Agent, A* Algorithm*

1. INTRODUCTION

In real life, when fire or some other uncontrolled situation occurs in places with a large crowd, chaos or even disaster may occur which often cause severe casualties. This is especially true in places where the internal structure is complicated and exit paths are limited which may incur serious casualties as people are unable to find the exits in panic. In such emergency situations, however, if people could be guided by some leaders who know the evacuation paths to the exits, casualties may be avoided or reduced significantly. For example, in some foreign countries, some staffs are assigned to important entertainment places to lead the people to evacuate from the danger in case of emergency.

One important issue for the managers of these entertainment places to consider is how many staffs need to be assigned to guide the crowd in emergency situations in order to maximize the efficiency of the evacuation process. In this paper, we will investigate how the evacuation efficiency is influenced by the number of customers and the number of staffs (or leaders) in the scenario.

We have simulated crowd evacuation from a dancing hall in case of emergency. The crowd consists of many groups, and each group includes a specific leader and some followers guided by the leader, who is responsible for path-finding for his followers. The objective of this simulation is to show the effect of different number of leaders on the efficiency of evacuation. A dynamic grouping algorithm based on the A * algorithm has been proposed to simulate the dynamic grouping phenomena with a number of leaders.

The remainder of this paper is organized as follows. Section 2 discusses related work in the field of crowd animation. The dynamic grouping algorithm is proposed in Section 3. The leader-follower model is discussed in Section 4. The simulation system is discussed in Section 5. Experimental results and analysis are described in Section 6. Section 7 concludes the paper.

2. RELATED WORK

The simulations of the group behavior, such as gathering, in emergency situation have been used widely in the group animations [1] [2] [3]. By setting some reaction rules in the emergency situation to the virtual characters, the gathering behavior of people in emergency situation can be simulated. However, it is not enough to simulate group behavior only by these simple rules, since human behaviors are much more complicated.

Cellular Automata technique has been employed by many crowd simulation models, especially in the field of evacuation simulation, including lots of researchers such as V.J. Blue, J. Dijkstra, A. Schadschneider, M. Fukui, H. Klpfel, P.G. Gipps, etc. and some commercial software such

as Egress and Exodus, for example, L.Z. Yang *et al.* apply a two-dimensional Cellular Automata model to investigate the advantages and disadvantages of evacuation caused by the kin behavior [4].

In recent years, many researchers make virtual crowd group animations by multiple level controlling [5][6][7][8]. In the area of the group animation, the interactive animation of leaders and followers is well developed [9] [10].

3. DYNAMIC GROUPING ALGORITHM

In the crowd simulation based on the *leader-follower model*, it is very important to model different groups in the crowd. For each group, there is a leader and a number of followers who follow the leader. Grouping can be divided to *static grouping* and *dynamic grouping*. Static grouping means that the crowd is grouped before evacuation, and every follower follows the given leader. Dynamic grouping means the members of each groups may alter during evacuation as the situation evolves.

Static grouping is suitable for situations where there is only one leader [9], but it is realistic for situations with many leaders. Since people are likely to follow the nearest leader which may change during evacuation, a dynamic grouping algorithm based on the A* algorithm is developed in our simulation system, which combines the process of path planning for followers and the dynamic grouping. The dynamic grouping algorithm is shown as following:

```
if(( the leader of current group arrives at the
exit&&newPath))||(number==0&&number !=0)){
   for (int i = 0; i<the number of grids ;++i){
     if (the current grid can be passed through){
       Calculate the value of H of patch[i] which is set to
         be the distance to the nearest leader for the path;
       Check out which the nearest leader is and set
         patch.group to be the number of the leader;
     }
   }
   Find out the location of the leader of the group based
     on A* algorithm;
   if (the leader arrives at the exit){
     Don't need to search and group; }
   Group crowd every 4 steps that the leader takes;
   Reset the number to 0 and restart counting;
}
```

Remark: the variable number indicates the number of steps the leader takes. The crowd is grouped every four steps in this algorithm. Function *FindPath* of class *PathFinder* is used to find out the path from current patch to the target based on A* algorithm; the value of index is set to 0 after every planning process, because the array path is used for storing then patch of new path which must be set to the new value. The Computational complexity of this algorithm is O(*kn*), where n is the number of grids.

4. LEADERS AND FOLLOWERS

4.1 Animation Model

The interactive animation of leader-follower behavior can be described with the following three levels: *action selection*, *steering* and *locomotion* [9].

Think about that a cowboy leads a group of cattle on the prairie. When a cow departs from the cattle, a messenger sends a message to the cowboy, and then the cowboy leads the cattle to the leaving cow. This massager takes the actor of *Action Selector*, which sends the information of changing direction to the cowboy; the cowboy act as a leader responsible for *Steering*, i.e., it will lead the cattle and avoiding collision. The cattle act as leaders which take simple *Locomotion*, i.e., following the leader.

In the three-level interactive animation model mentioned above, Path Finding and Following are the most pivotal factors that influence the simulation effect of the animation and the running velocity directly. In this paper, we developed a simple and efficient approach to implement the two pivotal factors using the leader-follower model.

4.2 Implementation of Leading-following Behavior

In our simulation system, the shortest path to the nearest exit is found for every leader as the evacuating starts, saving in the array *Path*. Select the Patch for next step from array Path before moving, until get to the exit. Every leader leads people to evacuate after followers have gathered around him, namely it needs a waiting delay.

At first, followers gather to the nearest leader. Different groups may have different leaders. When the evacuation begins, followers follow their respective leaders to evacuate to the exit. The path information for followers can only be obtained from the leader. During evacuation, followers do not follow the leader at the beginning - they need to determine for the nearest leader, and once they find a different nearest leader, they leave the current leader, following the new nearest leader. So the grouping process during evacuation is called dynamic.

5. DESIGN OF SIMULATION SYSTEM

This section introduces design of the simulation system to simulate the crowd evacuation with a leader-follower model. The A* algorithm is used for finding paths while the cell automata is used to represent the circumstance [11].

Considering the motion rules of crowd in real world, we summarize some basic concepts of crowd evacuation simulation system (based on two-dimension space) as follows: (1) Scene (Grid): it can be divided into some blocks, and can also be used to denote the simulation circumstance or scene. (2) Blocks in the scene (Patch): each block may contain some values to denote the state, such as whether it is occupie by somebody. (3) People in the scene (Person): this is used to represent the moveable individuals in the

scene. (4) Obstacle in the scene (Patch): it represents the fixed object that agents cannot pass through. It can be denoted by setting the states for Patch. There are three possible states: empty (value = 0), wall (value = 2) and person (value = 1).

Based on these abstractions, we can get an applied simulation model [12]. There are three pivotal classes. Therefore, we create these three pivotal classes: *Grid*, *Turtle* and *Patch*. Class Grid can be expressed as a two-dimension array, in which every element denotes a grid. Class Turtle represents agents which can move in the grids. Class Patch represents the value of grid. The pseudo-code can be shown as the following:

(1) Create a Grid object
(2) Initialize all panes in the grid
(3) Create some Turtle objects, and disperse them into the grid
(4) while (true)
(5) Call the update function for every pane
(6) Call the update function for every Turtle object
(7) end while

To show more details of the grid and make agents move, we add class *Display* and make every Patch object know the Grid objects nearby.

Based on the relationship between these pivotal classes, we can develop the class diagram of the crowd simulation system. Some screenshots from the simulation are shown in Fig.1, where the black objects represent wall and fixed obstacles, and the corresponding value for Patch is 2; the white ones represent the passable space, and the corresponding value for Patch is 0. There are two exits in the east and west, and the corresponding value for Patch of the grids outside the exits including the exits is 3. The moving red grids represent persons, and the value for Patch in which they stand is 1, assuming one grid can hold only one person.

6. RESULTS AND ANALYSIS

The scene of this system is a dance hall with two exits, in which there is a big dace pool in the center, with nine similar roomettes on each side. The red and green dots represent followers and leaders. The positions for the individuals are set randomly, and the positions of leaders are set nearer to the exits. As the simulation starts, leaders stand without moving, waiting for followers gathering. After the predefined waiting time T, leaders begin to move to one exit, and followers follow the leaders. During the evacuation, followers always follow the nearest leader, i.e., a follower may follow different leaders during the evacuation, as shown in Fig. 1.

Fig. 2 shows the function of evacuating time for different size of crowd, led by one to six leaders, where F is the number of followers. We can see that the evacuating time decreases significantly when the number of leaders increases from 1 to 2. However, as the number of leaders increases,

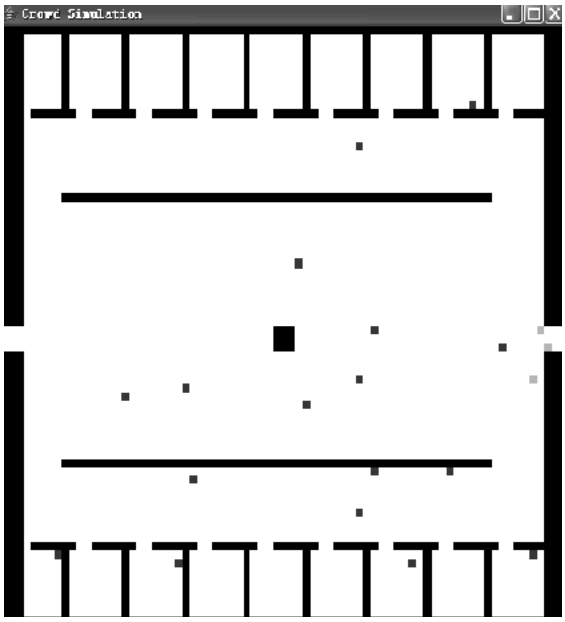

Figure 1: A Follower May Follow Different Leaders

the evacuation time increases steadily, and it even exceeds the evacuation time without any leaders: too many leaders result in bad effect to the evacuation.

At a first glance, it seems that more leaders will result in shorter evacuation time. However, our results show this is not true. This is because that too many leaders may cause chaotic effect to the evacuation. People often follow the nearest leader. As the number is large, people have to make more judgments and hesitate to move. There are two exits in the scene, and the evacuation time is shortest when there is one leader near each exit. When there are more than two leaders, chaotic effect occurs: some people may switch and follow different leaders during evacuation, hesitate to move, and it wastes a lot of time.

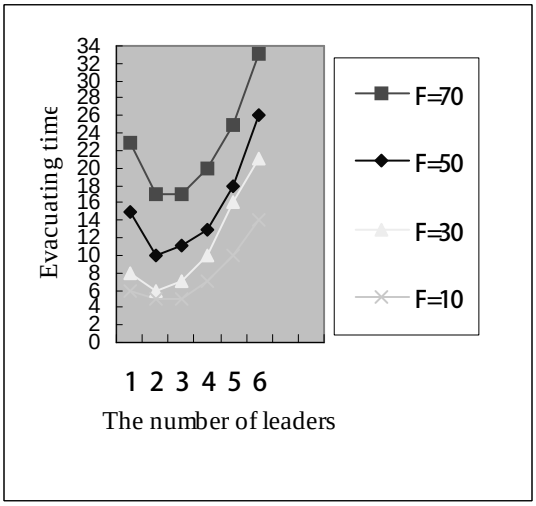

Figure 2: Experiment Results

7. CONCLUSIONS AND FUTURE WORK

We can get two conclusions from the previous analysis: (1) with some leaders, the evacuation will in general be more efficient as compared to situations with no leaders; (2) more leaders may not always result in more efficient evacuation; when there are too many leaders, the evacuation may take even longer time than that without a leader. Thus, an important question to ask is how many leaders will result in highest evacuation efficiency. It depends on the complexity of the scene, and there should be more leaders when there are more exits; if there are fewer exits, fewer leaders will be better. More research is certainly needed in this regard.

In our current simulation system, we just consider the position of leaders, ignoring other factors which may influence people's behavior during the evacuation, such as fear and congestion, and so on. In our future work, more factors should be considered for more realistic simulation effect. More over, the current scene is fixed with only two exits, different effects with different scenes still cannot be represented. We will investigate crowd behavior in more complex scenarios. The system is based on two-dimension cellular automata which has limited ability to represent the scene. We will extend the current system to three-dimensional virtual environment to present more realistic and immersive effect in the future.

ACKNOWLEDGEMENT

The authors would like to express their sincere thanks to Yanliang Liu, PhD Suiping Zou (NTU), Changbo Ji, Peng Yan, Jian Cao, and reviewers. This research is supported by National Science Foundation of P. R. China (grant No. 60473109), Science and Technology Tackling Project of Zhuhai Science and Technology Department (grant No. PC20061005), Open Project of State Lab of CAD & CG, Zhejiang University, and Guangdong Province Natural Science Foundation of P. R. China (grant No. 04300602).

REFERENCES

[1] C. Reynolds, "Flocks, Herds, and Schools: A Distributed Behavioral Model", in *Proceedings of ACM SIGGRAPH'87*, 1987, pp. 25-34.

[2] C. Reynolds, "Steering Behaviors for Autonomous Characters", in *Proceedings of Game Developers Conference*, 1999.

[3] X. Tu and D. Terzopoulos, "Artificial Fishes: Physics, Locomotion, Perception, Behavior", in *Proceedings of ACM SIGGRAPH'94*, 1994, pp. 43-50.

[4] L. Z. Yang, D. L. Zhao, J. Li, T. Y. Fang, "Simulation of the Kin Behavior in Building Occupant Evacuation Based on Cellular Automaton", *Building and Environment*, Vol. 40, 2005, pp. 411-415.

[5] S. R. Musse, D. Thalmann, "Hierarchical Model for Real Time Simulation of Virtual Human Crowds", *IEEE Transactions on Visualization and Computer Graphics*. Vol. 7, No. 2, 2001, pp. 152-164.

[6] T. K. Capin, I. S. Pandzic, N. Magnenat-Thalmann, D. Thalmann, "Integration of Avatars and Autonomous Virtual Humans in: Networked Virtual Environment", in *Proceedings of ISCIS'98*, IOS Press, Amsterdam, Netherlands, 1998, pp. 326-333.

[7] S. R. Musse, F. Garat, D. Thalmann, "Guiding and Interacting with Virtual Crowds in Real-time,", in *Proceeding of Eurographics Workshop on Animation and Simulation '99(CAS '99)*, Milan, Italy, Springer, 1999, pp. 23-34.

[8] I. S. Pandzic, T. K Capin, E. Lee, N. Magnenat-Thalamann, D. Thalmann, "Autonomous Actors in Networked Collaborative Virtual Environments", in *Proceedings of MultiMedia Modeling '98*, IEEE computer Society Press, 1998, pp. 138-145.

[9] C. W. Reynolds, "Steering Behaviors for Autonomous Characters", in the proceedings of Game Developers Conference 1999, California. 1999, pp. 763-782.

[10] T. Y. Li, Y. J. Jeng, and S. I. Chang, "Simulating Virtual Human Crowds with a Leader-Follower Model", in *Proceedings of the 2001 Computer Animation Conference*, Korea. (EI), 2001, pp. 93-102.

[11] P. Lester, "A* Pathfinding for Beginners", *http://www.gamedev.net/reference/articles/article2003.asp*

[12] Mitchel Resnick. Turtles, Termites and Traffic Jams-Explorations in Massively Parallel Microworlds. MIT Press, Cambridge, 1997.