AN OBJECT ORIENTED APPROACH TO SIMULATING AGENT-BASED BEHAVIOR

Sharad Sharma
sharad@umich.edu
College of Architecture and Urban Planning
The University of Michigan
Ann Arbor, Michigan 48109

James A. Turner
turner@umich.edu
College of Architecture and Urban Planning
The University of Michigan
Ann Arbor, Michigan 48109

Abstract

In this paper, we present an object-oriented approach for simulating agent-based egress behavior. The tool can be used when planners or designers need to assess the implications of their design or planning decisions. The tool is a web-based application that is being developed using standard protocols and technologies such as HTTP, HTML, and Java. It also demonstrates the effect of new building designs by simulating human evacuation behavior and relates the behavior of people. The existing models to describe the behavior of crowds usually deal with macroscopic variables like the average speed or the flow. Our aim is to model crowds as a malleable, moving masses, hoping that a more refined simulation might be achieved by considering each avatars behavior.

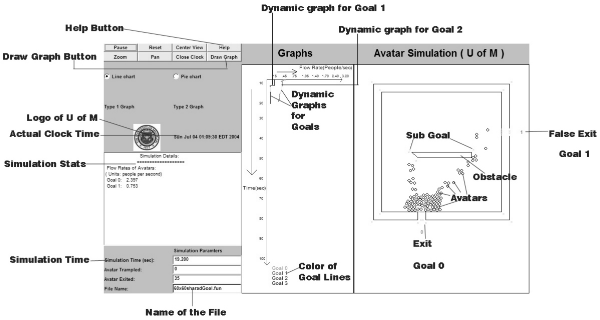
Key words: *Agent based simulation, Egress behavior.*

1. Introduction

Evaluating the characteristics of a crowd is a complex task. General surveys are made by asking people where they come from and where are they going and how long they walk and why they choose to stay at a particular point. Estimating crowd density is usually done from pictures and aerial photographs. Pushkarev in his research [1] made aerial counts of urban scenes involving 37500 pedestrians in New York. Panic situation analysis has motivated many research activities in the field of predicting movement of pedestrians (e.g. Helbing, 2000, Helbing, 2002). However, during a panic situation the behavior of individuals becomes irrational (Quarantelli, 2001) and the goal becomes as primitive as saving one's life. The behavior of pedestrians in architecture has been studied by Okazaki (1979), in urban planning by Jing (1999), and in GIS and agent based systems by Batty, Desyllas, and Duxbury (1999). With advancements in rendering technologies and virtual reality, simulation models with 3D representation have become popular. Agent-based approaches where the agent moves through a virtual environment have been studied by Penn and Turner (2002). Simulation of behaviors such as flocking has been widely used in creating realistic animations for group behavior using virtual creatures such as birds (Reynolds, 1987, 1999). Reynolds' computer model coordinated animal motion based on 3D computational geometry and is called as "boids."

1.1. Object-oriented programming approach to simulation

The proposed simulation uses an object-oriented approach with the main simulation functions and objects implemented in *Java*. The features of the graphical user interface are presented in Figure 1 and include the buttons, file name, U of M logo image, current time, current date, simulations statistics, simulation time, and dynamic graphs.



● Line chart
○ Pie chart

Sun Jul 04 01:09:30 EDT 2004

Simulation Details:
===================
Flow Rates of Avatars:
(Units: people per second)
Goal 0: 2.397
Goal 1: 0.753

Simulation Paramters
Simulation Time (sec): 19.200
Avatar Trampled: 0
Avatar Exited: 35
File Name: 60x60sharadGoal.fun

Figure 1: Features of the GUI

1.2. System Architecture

The Avatar Simulation is based on the interaction among objects constructed by the applet as shown in Figure 2. The system is divided into four phases: *Initialization, Creation, Simulation, and Ending*. When the applet is launched in a web browser, it automatically builds the objects of Avatar Simulation class. This is called the initialization phase of the system. This class in turn builds subsequent objects. The *Scenario* object represents the simulation engine and contains all the objects for creating the scene and contains *Person* object, *Wall* object, *Goal* object, *Scaled Canvas* Object, *Avatar Box* Object, and *Vector* object. An *Input File* defines the scenario object and the necessary parameters (walls, favorite goal, size, speed, wait time) for the simulation

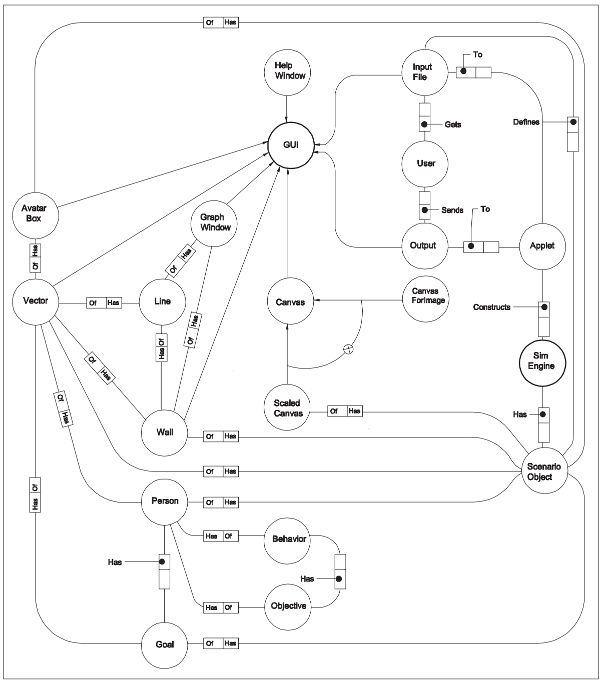


Figure 2: Simulation Conceptual model

The creation phase begins when the applet is launched. The *Scenario* object, *Graph Window* object, *Scaled Canvas* object, and *Help Window* object are created during the launch of the applet that displays the scenario in the web browser's screen. Each person has its own *Goal*, *Behavior*, and *Objective*. The avatars, walls and goals are displayed in the applet through the use of different colors. The simulation phase begins when the user presses the "run" button. The applet re-displays the scenario after every tenth of a second. Finally, in the ending phase, the user can see the displayed graph and simulation parameters on the applet.

2. Implementaion

The implementation of collision avoidance behavior of agents in the Avatar Simulation is based on a representation of trajectories in (x,y,t) space. By varying parameters such as goal selection wait time and steps taken between decision points, it is possible to generate aggregate movement levels similar to those found in an actual building context (real world). Helbing [7] has summarized the characteristics of "escape panic" by studying socio-psychological literature, reports in the media and available video materials [10], and engineering handbooks. In his study he states that in panic situations people tend to move considerably faster than normal and in an uncoordinated manner. At exit points, jams are formed that result in arching and clogging patterns. He also states that fallen people or injured people act as obstacles and alternative exits are overlooked or not efficiently used in escape situations. The Avatar Simulation model has adopted a similar approach where arching and clogging behavior is observed when jams build at the exits. The input file controls the maximum and minimum *speed* for each avatar. Similar to Helbing's model, the simulation also takes into account that interaction in a panic situation becomes physical, pushing behavior is observed and fallen people act as obstacles. The input file also controls the "*wait time*" when an avatar becomes aggressive and starts pushing other avatars and trampling occurs. At the start, the avatar does not immediately move towards the shortest exit. Instead it moves towards a favorite goal that is pre-assigned in the input file. In real time scenarios people tend to move towards familiar exits. The simulation also takes into account *false exits*, because in a real time scenario people tend to go towards a door that is not an exit. The avatars in the simulation move to another exit after reaching a false exit.

Collision detection is a geometric problem and has been extensively researched in robotics and computer-aided design. Collision detection fails when an object goes right through another object, while following a trajectory. If the agent takes a straight- line path to its destination, it will collide with the obstacle (refer path 3 in Figure 3). Alternatively, the agent can take a curved path around the obstacle. This is achieved by adding sub goals around the obstacle (refer path 1 and 2 in Figure 3). Figure 3 shows the interaction of an avatar with the obstacles in the simulation. Line 3 shows the path an avatar would take to reach a favorite goal. It is evident from the Figure 3 that the path to the goal is blocked by a wall and results in the avatar not "seeing" the goal. Line 1 shows the path, an avatar may use to reach a favorite goal. The simulation applet also displays a dynamic line graph (refer Figure 4) that plots flow rate (people/sec) vs. simulation time (sec) for each goal. An RGB color scheme is used for each goal to provide better visualization during the simulation. After the simulation ends the graph button can be used to visualize the bar graph and pie chart.

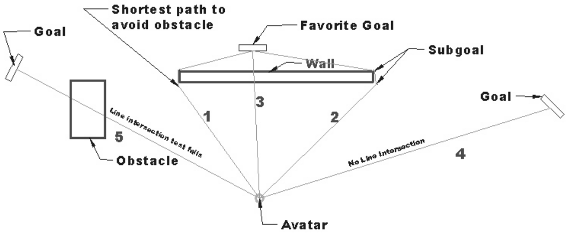


Figure 3: Interaction of avatar with obstacles

The scenario in Figure 4 depicts an evacuation behavior simulation for a lecture hall. The hall is 250 cm X 90 cm and has a stepped floor and a projector room at the back. For the simulation, the door to the projector room is as a false door. When the simulation starts, the goal of the avatars is to head towards the exits. In a real time scenario people have a preconceived notion of the exit they might use. To simulate this behavior each avatar is pre-assigned a favorite goal. If the avatars, on reaching the exit, find that there is not enough space to move, they become aggressive and start pushing each other. As the agents are placed randomly inside the room, the evacuation time is not same for all the two different simulation runs. Consequently, the graphs also show variations for each goal. The variation in the line diagrams is noticed in the Figure 4. The graph also shows that rate of flow at the false exit became constant after 15-20 sec. As initially people tend to move towards the false exit. The end of the line shows the time when the last agent exited from the exit.

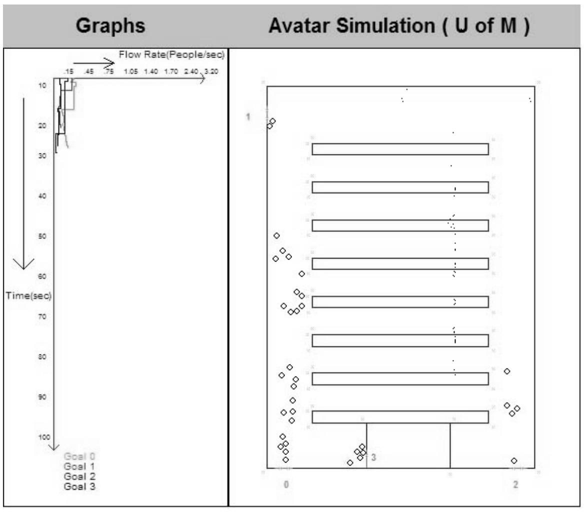


Figure 4: Screen shot of lecture hall simulation

3. Conclusion

The presented simulation model is based on a representation of space that makes it possible to consider both detouring and speed modifications in the avoidance patterns. Also, a sub-goal strategy is developed to allow people to effectively navigate around obstacles. If an avatars path to an exit is blocked, the avatar finds a sub-goal that is in line of sight with the avatar. The approach is general in the sense that differences in the scenario do not require modifications of the model but need only an input file to define the scenario. In this sense, the model is universal. The interactions with dynamic obstacles and static obstacles are taken into account. The simulation was tested in several sample situations and it was shown that the parameters of the algorithm could be adjusted to obtain a realistic collision avoidance behavior. However, several enhancements are to be implemented for validation before this model can realistically describe the complex patterns characteristic of crowd behavior.

References

1. Boris Pushkarev, with Jeffrey M. Zupan. Urban space for pedestrians: a report of the Regional Plan Association, 1975.
2. Helbing, D.; Farkas, I. and Vicsek, T. Simulating Dynamical Features of Escape Panic, Nature, v. 407, pp. 487-490, 2000.
3. James A. Turner, Conceptual Modeling Applied to Computer-Aided Architectural Design, Building Environment Journal, August, 1991.
4. James A. Turner, Guide to Reading NIAM Diagrams, PDES/STEP, 1988.
5. James A. Turner, A Case Study of Object-Oriented Design and Implementation of a Generic 3D Geometry Viewer," CAUP Web1996; September 1996.
6. Reynolds (1999) Steering Behaviors for Autonomous Characters, http://opensteer.sourceforge.net
7. Simulating dynamical features of escape panic, Dirk Helbing, Illeas Farkas & Tamaas Vicsek, Nature, 2000, Macmillan Magazines Ltd.
8. Quarantelli, E. L. (2001) Sociology of panic, in International Encyclopedia of the Social & Behavioral Sciences, Elsevier Science.
9. Okazaki, S. (1979) A study of pedestrian movement in architectural space. Part 1: pedestrian movement by the application of magnetic models, 283 111-119.
10. http://angel.elte.hu/panic/
11. Jiang, B. (1999) SimPed: simulating pedestrian flows in a virtual urban environment, Journal of geographic information and decision analysis, 3 (1) 21-30.
12. Batty M, Desyllas J, Duxbury E 2003 Safety in numbers? modelling crowds and designing control for the Notting Hill Carnival. Urban Studies, forthcoming.
13. Batty M, Jiang B, Thurstain-Goodwin M 1998 Local Movement: Agent-Based Models of Pedestrian Flows. Working Paper 4, Centre for Advanced Spatial Analysis, University College, London.
14. Behavioral dynamics for pedestrians, Bierlaire, M., Antonini, G. and Weber M., International Conference on Travel Behaviour Research,August 10-15, 2003.