Believable and Reactive Crowds in Next Generation Games

Paul A. Kruszewski

CTO, Engenuity Technologies, Inc.
4700 de la Savane, Suite 300
Montreal, Quebec H4P 1T7 Canada
paul.kruszewski@engenuitytech.com

Abstract

The creation of life-like believable characters is emerging as the central focus of next-generation game development and is viewed as critical to obtaining true mass-market appeal. Reactive crowd simulation is a critical component in creating believable character. The demo runs on a Xbox 360 and shows the following crowd behaviors. A typical downtown is populated with pedestrians and vehicles. The pedestrians mill about the town staying on the sidewalk and cross only at street corners (although some decide will "jay walk" from time to time). Occasionally, they will stop to talk with one another. Cars drive around the city and obey the traffic lights. The ambient behaviour is interrupted by traumatic event; in this case a terrorist pulls out a run and starts firing at the crowd who now move into a panic behaviour. Some crowd members run away immediately from the terrorist while others first cower in fear before running away. Crowd members who do not hear the weapon fire but see a panicking pedestrian may also panic or continue about their business depending on how "cool" they are. Vehicles also flee the terrorist and will drive on sidewalks and over pedestrians to save themselves.

Introduction

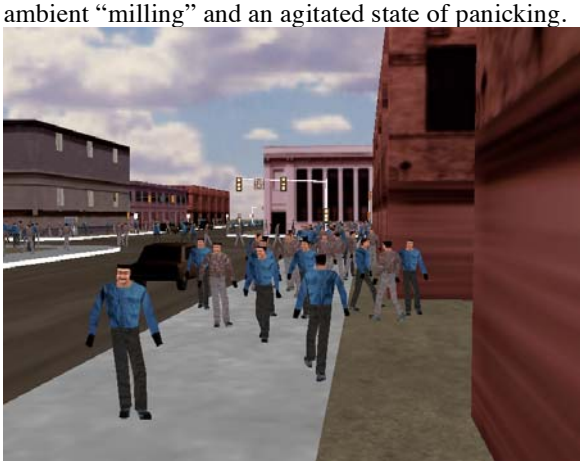
Next-generation gaming hardware such as the Xbox 360 and PlayStation 3 will allow the creation and visualization of large visually rich but virtually uninhabited cities. It remains an open problem to efficiently create and control large numbers of vehicles and pedestrians within these environments (Crance 2006). We present a system originating from the special effects industry, and expanded in the military simulation industry, that has been successfully evolved into a practical and scalable real-time urban crowd simulation game pipeline with a behavioral fidelity that previously has only been available for non-real-time applications such as films and cinematics (Kruszewski 2006).

Goals and Functionalities of a Crowd System

Whereas traditional game AI is designed to interact directly with the player, crowd AI is first and foremost part of the background. In special effects, it is commonly called a dynamic matte painting. Indeed, crowd AI can be thought of as the AI level-of-detail equivalent of a sprite. As such, the number of crowd members is more important than the intelligence of the individuals. As moving matte paintings, the most common functionality is to have the crowd milling about. That is, pedestrians that can walk along the sidewalks without running into each other and cross at the cross walks, and vehicles that drive on the road and obey stop signs and traffic lights. In almost every game genre, this ambient environment is periodically interrupted by some kind of special (typically traumatic) event, ranging from a touchdown by the home team to an explosion, to which the crowd must react to accordingly. This reaction can range from "doing the wave" around a stadium to stampeding away from gun fire.

Demonstration of the system

We use as our test bed a small city called SimTown. SimTown is 1 km by 1 km and contains 93 buildings as well as numerous streets and sidewalks. All art assets, including SimTown, the character models and animations were generated in Maya and exported to our custom demo engine called Arena which was built using EGT's GameBryo middleware for rendering and character animation. The scenario has two acts: a normal state of ambient "milling" and an agitated state of panicking.



In the normal state of ambient "milling," pedestrians stay on the sidewalk and cross at the crosswalks, and vehicles

stay on the road and obey traffic lights.
Transition from the normal state to the panic state is caused by a "traumatic event," in this case a gun being fired in a crowded street.

In the agitated state of "panicking," pedestrians run away from the traumatic event (a terrorist firing a weapon) and will run across street while trying to avoid collisions with other pedestrians and vehicles.

Normal Milling State

In this example, we want to simulate normal traffic in a busy city of cars driving around and people walking about.

Vehicles obeying traffic rules. We simulate the vehicle driver by an autonomous character that controls the position and orientation of the vehicle model. To generate traffic circulation, each driver chooses a random destination from a set of pre-assigned waypoints. This has the added advantage of allowing the level editor to directly control the distribution of the traffic within the city. The road network is represented by a waypoint graph. The driver uses it to find a path between itself and its destination. Once it arrives at the destination, it chooses a new destination. The directed edges of the waypoint network ensure that that the cars stay in their lanes.

In order for cars not to rear-end one another, decision making logic must be added to allow them to sense other vehicles. When a vehicle ahead is being approached, the car slows down via queuing collision avoidance. Collisions at intersections are avoided in by implementing stop signs and traffic lights. Each traffic light is implemented as an autonomous character that cycles through the states of green, yellow and red. Each driver has a traffic light sensor that allows it to see the traffic light's state. If the driver sees a red light and it is near a stop point, it will stop. Stop points are located just before crosswalks. Cars behind the first stopping car stop due to queuing collision avoidance. When a driver sees a stop sign, it will go directly to the stop point in the street. There it will ask an invisible traffic cop for a ticket to pass through the intersection. When its ticket becomes active, it will proceed through the intersection. Once it has cleared the intersection, it returns the ticket back to the traffic cop. This ticketing system allows for irregular stop sign configurations such as three-way stops.

People milling about on sidewalks. Just as cityscapes are full of cars driving around, people mill about on sidewalks, crossing at cross walks. Since the movement along a sidewalk is significantly less delineated than a street, it is more appropriate to use a navmesh for navigation. Similar to a driver, a pedestrian will choose a destination at random. The pedestrian uses the navmesh to find a path between itself and the destination that only involves only sidewalk or crosswalk cells. Once it arrives at the destination, it chooses a new destination. Collision avoidance prevents pedestrians from running into each other. Blind data restricts street crossing to crosswalks. Vision sensors and decision trees prevent pedestrians from crossing against the traffic lights.

Agitated Panicking State

In our example, a terrorist can be controlled to walk around the crowd. The user can trigger the terrorist to pull out his weapon and start firing. At this point, the terrorist puts himself into group of agitated characters. A crowd member will go into the agitated state if he sees any members in the agitated group nearby. This technique causes the panic to ripple outwards from the terrorist towards crowd members who do not directly see the terrorist. When a crowd member goes into the agitated state, it chooses a new target destination that takes away from the traumatic object. The pathfinder now ignores blind data and chooses the shortest path, thereby potentially causing the crowd member to cross the street.

References

Crane, S., "Bringing Game Characters to Life," *Journal of Game Development V1(3)*, Charles River Media, 2006: pp. 51 – 61.

Kruszewski, P., "Real-time Crowd Simulation Using AI.implant," *AI Game Programming Wisdom 3*, Charles River Media, 2006: pp. 233 – 248.

Acknowledgements

The AI.implant system is the result of many years of hard teamwork. The author would like to thank the entire AI.implant development team at ETI for its ongoing excellent work, and in particular Cory Kumm who developed the .ACXs and artwork.