CrowdViewer: From Simple Script to Large-scale Virtual Crowds

Tianlu Mao
Bo Shu
Institute of Computing Technology, CAS
Graduate School of the CAS

Wenbin Xu
Shihong Xia
Zhaoqi Wang
Institute of Computing Technology, CAS

Abstract

Visualization of large-scale virtual crowds is ubiquitous in many applications of computer graphics. For reasons of efficiency in modeling, animating and rendering, it is difficult to populate scenes with a large number of individually animated virtual characters in real-time applications. In this paper, we present an effective and readily usable solution to this problem. It accepts simple script which includes motion state and position information of each individual at each time step. Supported by material database and motion database, various human models are generated from model templates, and then driven by an agile on-line animation approach. A developed point-based rendering approach is presented to accelerate rendering. We test our system with script including 30,000 people evacuating from a sports arena. The results demonstrate that our approach provides a very effective way to visualize large-scale crowds with high visual realism in real-time.


Keywords: virtual crowds, modeling, animating, rendering

1 Introduction

Vivid virtual crowds are vital for many applications using computer graphics techniques for example, games, movies, applications of public safety, transportation and education. In some cases, large-scale virtual crowds which include thousands of, or even ten thousands of virtual characters need to be simulated such as simulating thousands of spectators evacuated from a sports arena, passengers in an underground railway station, large-scale battle scenes and so on.

Crowd simulation problem can be solved by dividing it into two sub-problems. The first one is planning which generally includes path planning and choosing appropriate motion state for each character along the path. The other sub-problem is visualization. That is to create 3D models of virtual crowds, animate each individual according to given path and motion state, and then render the dynamic crowd scenes. The methods for planning depend on the specific requirements of the corresponding application. In some network games, paths and behaviors may be decided by network players. In applications of public safety, such as evacuation simulation, plans are made automatically by special software and many social psychology factors need to be considered in decision models. However, the methods for visualization are universal to some extent. They generally include 3D geometric modeling, animating and rendering steps. Recent advances in computer graphics techniques and graphics hardware made it possible to visualize a small crowd with high visual realism. Unfortunately, problems become tougher when the number of involved characters in crowds increases to more than thousands while the real-time results are expected at the same time.

Bottlenecks are obvious under the double restrictions - real-time and the large number of characters. First, it is too costly and even unpractical in some applications to create 3D model and motion data for each virtual character. Second, computational resources for each frame are limited in real-time applications. Therefore, simpler approaches to animate virtual characters individually are needed. Third, optimizing algorithms for rendering thousands of characters are also needed due to the same reason.

Although no complete solution is presented, there are still some works concerned with it. Most of them focus on rendering [Tecchia and Chrysanthou 2000] [Dobbyn et al. 2005] [Wand and Straßer 2002]. Additionally, in [Ulicny et al. 2004], the question of how author crowd scenes in a simple and intuitive way is investigated, and an interactive tool named Crowd Brush was produced to create new individuals and change their appearances or behaviors. Pre-computed deformed meshes of each animation frame were stored for each model template to accelerate rendering. [Ciechomski et al. ] improved this method with 4 static levels of detail meshes and received good results. In [G.Ryder et al. 2005], a framework of navigation and interaction system of virtual crowds is introduced which could sustain about 1,000 characters in real-time. In [de Heras Ciechomski et al. 2006], a scalable simulation framework of virtual crowds is introduced which could sustain about 10,000 pedestrians at about 9 fps. EXODUS is a serial software for evacuating simulation [the University of Greenwich. a]. It provides a tool named v EXODUS to create animated three dimensional representations of results generated from EXODUS [the University of Greenwich. b]. However, the number of characters visualized in real-time is limited according to their demonstration.

In this paper, we focus on the sub-problem of visualization in crowd simulation. The motivation of our work is to try to design and implement a complete and universal framework for dynamic crowd visualization under the restrictions of real-time and large crowd of ten thousands individuals. There are two main contributions in our work.

First, the framework is universal to almost all applications related to crowd simulation. The input of our visualization framework is general and simple which contains scripts describing motion state and position information of each individual at each time step. By this way, the system provides flexible and convenient interface for manipulating virtual characters. It can be easily embedded into or invoked by other software to enable real-time dynamic crowd visu-
alization.

Second, we developed an effective approach that enables visualization of large-scale virtual crowds at interactive frame rates. What we do mainly is not to improve existing algorithms of modeling, animating and rendering, but to select them strictly and make a perfect trade-off between visual realism and efficiency.

Additionally, we have developed a software named CrowdViewer under our framework, and applied it to actual projects of public safety to visualize the results of evacuating simulation. Experiments show that our system provides a very effective way to visualize more than 30,000 virtual characters with high visual realism at interactive frame rates.

2 Overview of our framework

The goal of our work is to develop an effective and readily usable framework for real-time visualization of large-scale virtual crowds. An overview of the framework is given in Figure 1.

**Input:** The input of the framework is simple scripts containing information of there aspects of each individual: model type, position and motion type in each time step. A small segment of an example script is given as below:

```plaintext
Person ID: 17;
Person type: young man;
Planning results:
t0: position:(0,1,1), motion type: looking around;
t1: position:(0,1,1), motion type: walking;
t2: position:(0.2,1,0.1), motion type: walking;
t3: position:(0.4,1,0.2), motion type: walking;
```

**Preprocessing:** In order to sustain simple input and speed up run-time processing, model templates, material database and motion database need to be created in advance. Usually, individuals in virtual crowds could be classified into different types by criterions such as age, gender, occupation and so on. In preprocessing step, model templates need to be created for each kind of model type. Afterward, for each template, corresponding material database need to be created. At run time, the templates could be easily modified by applying different materials from corresponding material database. And also, corresponding motion database which includes periodical motion data for different motion state such as idle, walking and running, are created for each template. The specific contents of databases depend on the specific demands of the application.

**On-line Processing:** At run time, first, under the assistances of model templates and material databases, model variation technique is used to model various virtual characters. Then, supported by motion databases, an agile animation technique is used to generate vivid animation for each character individually according to motion state and position information inputs. Finally, a developed point-based rendering technique is used to enable real-time rendering of large-scale dynamic crowds.

3 Modeling

Under the double restrictions: real-time and the large number of characters in crowds, there are two difficulties need to be solved in modeling step. One difficulty is to make the process of geometric modeling as automatic as possible. The other is to decrease the memory storing these models to such extent that normal PC can afford. Therefore, the best way is that models of different characters share the same data and make some fast variations at run time so that characters look different.

In [de Heras Ciechomski et al. 2005], virtual humans with variety in color, animation and appearance are generated by using vertex and fragment shaders. Since there’re still many GPU that are not programmable, we developed a model variation algorithm based on model templates and material databases in our framework.

For each model type, a model template and corresponding material database are created in pre-processing. A template is a skeletal 3D model with a skinned mesh which could be easily deformed by changing the joint angle of the corresponding skeleton. For each model template, several significant parts of mesh are selected, such as T-shirt, trousers and so on. Then, for each significant part, some candidate material ingredients are stored in material database.

At the beginning of visualization, for each character, the system chooses a model template according to its model type. Then material ingredients are randomly selected from the corresponding material database. In rendering, the selected material ingredients are used to substitute the original material on the selected template. Figure 2 shows variations from one model template. Thus, characters belonging to the same model type share the same template. The only difference among them is the index of material ingredients. The increased memory cost for one additional character will be very subtle, and the data of each model type including template with medium specific, motion database and material database is about 17M.
4 Animating

Since a large number of characters need to be animated in real-time, periodical motion data need to be synthesized in advance, which could be displayed periodically to generate continuous motion. We get original motion data from motion capture device at high sampling frequency, and then synthesize periodical motion data by motion editor software. At run time, system selects motion data from motion database according to the model type and motion state of the character.

Since the velocity and direction of the character may change at run time, selected motion data need to be re-sampled and re-directed. In our framework, we developed a motion deformation algorithm.

We define the motion data deformation problem as follows: assume $F$ as a periodical motion data. $F_i$ is the $i_{th}$ frame data and $R_i$ is the root position of $F_i$. At the simulation time step $j$, assume $P_j$ as the planned root position of a character and $F_{k_j}$ is the selected motion data from F. The motion data it used can be describes as $F_{k_j} + (P_{j+1} - P_j)$, where $(P_{j+1} - P_j)$ is the variety of the direction of root. What we want is to find a $F_{k_{j+1}}$ in $F$, which let the character moving smoothly to $P_{j+1}$ at the next time step. We use a trial based method, search in $F$ orderly from $F_{k_j}$ until we find a frame which make $||R_{k_{j+1}} - R_{k_j}||$ is the closest to $||P_{j+1} - P_j||$.

The motion deformation algorithm is simple but very agile. When a character maintains in one motion state, the animating results look natural and smooth, and only very few subtle footskate could be found. That partly profit from the high frequency of the motion capture data. The data we use is in 120Hz. However, when a character transfers from one motion state to another, things become tough. Up to now, no motion synthesis algorithm is efficient enough to sustain ten thousands characters under real-time restriction and providing smooth transition between two motion states. Therefore, simple linear interpolating algorithm is adopted in our framework temporarily.

5 Rendering

There are two categories of rendering technique which could be adopted to speed up large-scale dynamic crowd rendering: image based rendering and point sample technique.

In image base rendering, 2D image-based representations (i.e. impostor) are used to accelerate rendering. Thus, maintain thousands characters at interactive frame rates [Tecchia and Chrysanthou 2000]. However, it needs great amount of memory to store textures for each individual in each possible pose and each possible viewpoint. In [Dobben et al. 2005], real-time impostor shading and texture thrashing reducing techniques were introduced to improve the results. However, when we increase the number of model templates and the size of motion database, great amount of memory is also needed and texture thrashing will still become a problem.

In our framework, we have developed Wand’s approach by abandoning hierarchical instantiation technique and accelerating rendering method to support individually animation in large-scale crowd.

When we tested the approach in [Wand and Straßer 2002], we found that after sampling the coarsest model generated for one virtual human model is composed of dozens of triangles and hundreds of points, which are unnecessary when the projection of model contributes only a few screen pixels to the final image. We modified the sampling step by doubling the border of the root bounding box of the octree then put the model template at the corner of root bounding box. As a result, the coarsest detail is reduced to only dozens of samples while only one node will be added on the top of the octree. Table 1 shows the coarsest representation of six human model by Wand’s approach and our approach.

<table>
<thead>
<tr>
<th>Number of the Coarest Models in Crowds</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>by Wand’s</td>
<td>205 p</td>
<td>193 p</td>
<td>199 p</td>
<td>169 p</td>
<td>206 p</td>
<td>218 p</td>
</tr>
<tr>
<td>by us</td>
<td>42 p</td>
<td>45 p</td>
<td>45 p</td>
<td>42 p</td>
<td>43 p</td>
<td>50 p</td>
</tr>
</tbody>
</table>

After sampling, for the purpose of rendering acceleration we generate a multi-resolution octree structure for each model template and rearrange the whole tree in depth-first order. Then, each octree node is packed into a 32-byte structure. The primitives within each octant are collected into an array and placed in local memory of the graphics card.

We tested our method by rendering a scene including 30,000 virtual characters while changing the proportionment of models presented by the coarsest detail. Because of the huge number of characters, most of them are far away from the viewpoint and contribute only a few screen pixels. The rendering approach is accelerated since we reduced the redundant information in the coarsest representation. Figure 3 shows comparison of performances between Wand’s approach and ours. Experiments shows we got more than 15 percents acceleration.

![Figure 3: Time consuming for rendering one frame of 30,000 virtual characters by Wand’s (blue line) and ours (red line).](image-url)
6 Experiments and results

Under the framework, a software named CrowdViewer is developed and released. We applied it to actual projects of public safety in which large-scale crowds evacuated from a sports arena and underground railway station were investigated. The planning results were generated by researchers in public safety field using professional evacuation software EXODUS and were transferred into text script file as input. For such application, we divided characters into six categories according to their age and gender. They are old man, old woman, young man, young woman, girl and boy. Therefore, six model templates were created. Corresponding material databases were created with colors randomly generated. Motion data for behaviors under situation of evacuation were captured. They are looking around, walking, running, going upstairs and downstairs.

We tested our system with dynamic virtual crowd including 5,000, 10,000, 20,000 and 30,000 persons. All of our tests were performed on a PIV 3.4Ghz processor, with 1G RAM and a GeForce 6600GT 3D card with 128MB video memory. The efficiency of our visualization system is shown in Table 2. Figure 4 shows details of our visualization results.

<table>
<thead>
<tr>
<th>Speed (fps)</th>
<th>5,000</th>
<th>10,000</th>
<th>20,000</th>
<th>30,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>62.50</td>
<td>32.25</td>
<td>15.87</td>
<td>9.17</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Efficiency of our system.

7 Conclusion

In this paper we presented an effective and readily usable framework for real-time visualization of large-scale virtual crowds. Script which describes motion state and position information is adopted as input, provides convenient interface and makes our framework universal to almost all application of crowd simulation. Existing techniques in virtual crowd modeling, animating and rendering have been combined and improved. Thus, dynamic virtual crowds are generated from simple script under the rigorous restrictions: large crowd of ten thousands individuals and real-time limit.

We also developed practical software- CrowdViewer under the framework. We are so encouraged that our software has been applied into actual projects of public safety to help them visualize the simulation results of evacuation. According to their feedback, our method appears to be uniquely attractive for a number of reasons: It sustains more than ten thousands individuals in real-time; it provides convenient interface which can be easily combined into workflow with other software; it has high visual realism that makes research, training and demonstrating more intelligible.

Acknowledgements

The authors would like to thank researchers from China Academy of Safety Science and Technology (CASST) and Beijing Municipal Institute of Labour Protection, Prof. Hui Ding, Prof. Maohua Zhong, Dr. Wei Li and Dr. Chongling Shi for providing scripts in this work. This work was supported by National Natural Science Foundation of China granted 60573162, 60533070, 60603082 and 60403042. Also, by Beijing Natural Science Foundation of China granted 4062032, National 863 Foundation of China granted 2006AA01Z336 and 2007AA01Z320.

References


