The Knowledge Engineering Review, Vol. 23:4, 399–412.
doi:10.1017/S0269888908000052 Printed in the United Kingdom

Art applications for crowds

ISAAC RUDOMIN, ERIK MILLÁN, BENJAMÍN HERNÁNDEZ, MARISSA DÍAZ and DANIEL RIVERA

ITESM-CEM, Carretera lago de Guadalupe Km. 3.5, Atizapán, México;
e-mail: rudomin@itesm.mx, emillan@itesm.mx, hbenjamin@itesm.mx, dmarissa@itesm.mx

Abstract

We have developed two different art applications as a way of guiding and testing of our technical work in crowd simulation and interface devices. In this paper, we describe both of them as well as the systems used to implement them. 'Crowds' is an immersive art installation with stereo projection based on our development of the specification of crowd behavior using XML and images, a proprietary tangible wind interface, and interactive real-time navigation. 'UnderCrowds' deals with large crowds, implemented in the graphics processor, and uses a crowd sensor to make the size of the virtual crowd proportional to the number of spectators crossing in front of the said sensor.

1 Introduction

Virtual art, a kind of art that uses interactive graphics via interfaces, is going through a state of plurality. In the last decade, traditional virtual reality reached the level of compatibility between the interface and the graphics. There came to be a distinction between immersive art—mixed reality, CAVE—and interactive art, mostly that without stereo projection. Oliver Grau (2003), in his book 'Virtual art: from illusion to immersion', has an entertaining and complete account of the story behind virtual art. There, among other important work, the work by artists such as Charlotte Davies, Christa Sommerer and Jeffrey Shaw is explored. He considers the work of Jeffrey Shaw and Charlotte Davies as the most immersive, since spectators could use interfaces capturing their movement. At the same time, the subtleness of the interfaces does not interfere with the contemplation of graphics and animations. Virtual art includes not only that using stereo glasses but also the real-time response of the system. Work by Christa Sommerer (with Laurent Mignonou) is an important part of this repertoire. In our case, it is interesting to focus on the nature of Charlotte Davies's work. It is particularly important for its use of natural user interfaces in a naturalistic world. Her work with interfaces for virtual reality in 'Osmose' is yet to be equaled in the degree of immersion and the finnesse of the interface. This work, however, required the use of what was then very expensive equipment.

The high cost of producing such work made this type of art exclusive of a few well-financed artists. In addition, the most transcendent premise in electronic art is the human/computer relationship. Interfaces would be in the center of expectations. Ishii introduced the idea of tangible interfaces, blurring the boundaries between the real and the virtual. Thinking that it was time to do something similar, and commissioned by the late Priamo Lozada, we developed 'Fluids', an installation that was presented in 'Dataspace', an exhibit he curated in Madrid 2005. 'Fluids' introduced tangible natural interfaces as a way of integrating the spectator in the immersive experience. Touching water and breathing redefined the virtual landscape. Nature was incorporated to technological art, because at the end of it this is what one looks for in the esthetic experience: an exploration of the senses.

Much like Char Davies's (2003) 'Osmose', we developed an interactive and immersive mixed reality art installation. Its purpose was to explore the relation of intimacy between reality and virtuality by emphasizing naturalistic environments, intuitive navigation, and using air and water as interfaces. Of course, given the advance of technology, we designed the environments and the installation so as to take advantage of the low-cost alternatives available. But in contrast with her work, 'Fluids' combines interaction with immersive mixed reality with the purpose of blurring the distinction between the realities.

Our development is of low cost, showing that there is an alternative for this kind of development. This piece is described in more detail in Rudomin *et al.* (2005a) and Díaz & Rudomin (2005).

'Fluids' was very successful, but even with all the effects one could do, and even with very effective if maybe not fully exploited interfaces, the virtual worlds we could visit remained mostly lifeless, desolate spaces, navigated by lonely visitors. Our goal then for our next project was to populate these spaces, and have them be occupied by crowds with behavior simulating that of 'real' reality. This was possible because we are using art applications as an ideal testing ground, allowing us to deploy and guide our developments in software implementing crowd behaviors, as well as custom hardware for intuitive user interaction.

As time went by, 'Fluids' morphed into 'Crowds'. If something had characterized virtual reality it was the desolate graphics spaces, even if they reached a high level of esthetics. Incorporating crowds in a party in an abandoned building, and giving them the schizoid tone of videogame characters, made the piece more like cinematography.

'Crowds' is an immersive art installation with stereo projection based on our development of the specification of crowd behavior, by using XML and images, a proprietary tangible wind interface (similar to the one used for 'Fluids'), and interactive real-time navigation. 'Crowds' was built taking 'Fluids' as a basis. Some sensors originally developed for 'Fluids' were further developed and recast for different uses in this piece. However, the main purpose of this new piece was to allow the artist to construct scenes populated by different characters. In addition, interaction using these devices can take place. An important difference is that in the 'Crowds' system, the artist can specify simple narratives within the scenes by writing a script describing the activities of the characters within the scene, transforming the visual artist into a kind of virtual film director, but retaining the interactivity of the scenes.

The underlying technology that was used for implementing 'Crowds' was XML. Visuals slid into interactive cinema, a practice that still happens seldomly. In this work, the movie director's role is taken by the visual artist generating a script. It is like reencountering the origin of virtual reality: cinerama. Even though there is no dialog, in this work two moments of images are in contraposition: black and white film and videogames, in a kind of pseudorealism.

At that time, we were working on a different project involving the use of the graphics processor (GPU) for simulating large crowds. When this project started to give us important results (an order of magnitude larger crowds than those that could be simulated with competing systems), we decided to use this on a new project that took on the name 'UnderCrowds'. This is an art installation dealing with large crowds. We had to rewrite the crowd simulation system used in 'Crowds' because it could not deal with large numbers of characters in real-time. We developed a new and innovative system, where both the behavior of the characters and their rendering are implemented in the GPU. This system allows us to simulate and render, in real-time, crowds of several hundred thousand characters. Behavior definition uses maps and scripts in a manner similar to that used for 'Crowds', but in this case this specification is translated to custom shaders that are executed within the GPU.

But crowds have a specific field that has been explored seldomly in interactive graphics. Outside videogames, in virtual art there have been no reports of artists working on the subject yet, maybe because we are used to treating this subject in a social and political context. This took us to a new development: an interface capturing crowds in streets, the subway and public plazas to represent them in real-time. The system reports approximately how many people are passing and draws them in real-time in the screen. This required us to develop a new sensor for interaction, namely a crowd sensor that was used to make the size of virtual crowds proportional to the number of spectators crossing in front of the sensor. The individual or individuals captured become part of the world of the crowds: following a leader, dispersing, etc. There is no contact back to the real world, but with a naturalist, biology simulation. A spectator of this piece only sees the behavior of enormous crowds, reaching the desired esthetic level due to the great number of characters that the new system makes possible.

'Crowds' was shown in IVEVA 2006 in Aguascalientes, although without the sensors. 'UnderCrowds' was developed for (and shown in) TECHNARTE 2007 in Bilbao.

In the following sections we describe each piece in more detail. In each case we first touch on the technical aspects, and then go on to describe how these techniques were used.

2 Crowds

'Crowds' is populated virtual immersive space. What previously had been desolate spaces are occupied by crowds, with behavior simulating that of 'real' characters. In this piece we take into account visual tradition, generated by film, photography and video, but now technology allows us to generate nonlinear narrative for interactive environments. Graphically, the piece makes allusion to black and white films, antique photography, and the desolate landscape of a bygone city of which only a single building is left as a quote from the past.

The breathing pattern of the user is detected by a heat sensor, which is used to generate weather (snow), moved by the user's exhalations that also modify the fog. In other scenes, the breathing pattern of the user can change the chromatic gamut of fantasy cities where characters seem to live. Just by blowing, the facades of buildings change. 'Crowds' can be thought of as a flexible art piece that can take us closer in the understanding of crowds.

2.1 Specifying behavior with XML and images

Defining animated characters that interact with the user, with other virtual characters, and with their surrounding environment has been approached in many different ways. Work in this area is extensive, starting with boid emergent behavior (Reynolds, 1987), use of finite state machines (FSMs) (Cremer *et al.*, 1995) and hierarchical schemes (Musse & Thalmann, 2001), among many others.

Extensible markup language, or XML, is a simple, flexible text format used in many domains. It has been used by different researchers for specification of character behavior (Gebhard *et al.*, 2003). In our paper (Rudomin & Millán, 2004) we describe the behavior of virtual characters and groups by using XML scripting and images. 'Crowds' is based on this development.

A simple way to specify the behavior for virtual characters is through the use of FSMs. A set of XML tags can be defined to specify state diagrams. Without going into details, we have the following: agent types are defined by using the `<agenttype>` tag, which contains a set of states describing their behavior. Each state, defined through the `<state>` tag, has an associated name, which identifies the state for transitions from other states. Every state contains a set of atomic behaviors. These behaviors are specified by the `<behavior>` tag, and refer to simple actions. The set of atomic behaviors within one state is executed sequentially. In order to consider a behavior as completed, it must satisfy a condition; for instance, the maximum time allowed for that behavior, or reaching a location previously specified. Once completed, the agent proceeds to the next behavior in the state, or, if the executed behavior is the last in the state, to the first behavior.

A special type of behavior is a transition, defined by the `<transition>` tag. The required parameter for a transition is the name of the state that the agent will switch to. After definition, agents can be created either individually or in group. The `<agent>` tag specifies an agent by providing its agent type and location and a geometry file to display the model. It is also possible to specify a group of agents.

In virtual environments with many virtual characters, it is important to provide simple mechanisms for specification of character behavior, environmental properties, and the interaction between them. A simple way to define information related to the scene is image maps. In Rudomin *et al.* (2004), we explored the use of image maps and found out that they are intuitive for a user, and can be easily edited, stored, and loaded. Our system uses images as auxiliary maps for the definition of behavior of agents. In Rudomin & Millán (2004) we combined maps and XML scripts. Maps are defined in the XML script using the `<map>` tag. We can define several kinds of maps. Some (path maps, height maps, texture maps, label maps) are static and can be loaded by the user, or updated interactively. Other maps are dynamic (collision, visibility, level of detail) and can be used by the system if so desired. The combination of scripting and FSM's as XML scripts proved very fruitful.

In Rudomin *et al.* (2005c), we further developed the idea by including probabilistic, hierarchical and layered FSM's to our system. Probabilistic schemas increase the apparent diversity of behaviors, as the behavior for individual characters is not uniquely guided by rigid scripts, but rather modified in unexpected, yet controlled ways, by randomly selecting certain transitions within the FSM.

2.2 *Wind interface*

Mixed reality and tangible interfaces have been proposed to fill the gaps between virtual worlds and the tangible, physical world. Human computer interface (HCI) has changed throughout the years, and evolved to integrate reference objects to improve interaction. In our work, we want the user to experience a more natural, friendly interaction. Tangible interfaces have been shown by Ishii (1999) to make the user feel more connected to the virtual world by using real-world objects as interfaces. Ishii also uses natural elements such as sand, air, and water as reinforcing elements for interaction.

In virtual reality applications, the use of tangible interfaces allows the user to get involved with the virtual world in a more fluid and explicit way. 'Crowds' uses a proprietary wind interface that uses a device that detects the changes of air temperature caused by breathing, that we developed for our previous work 'Fluids' and have since improved. It is a 'wind' or 'breathing' interface that uses a temperature-sensing device connected to a sound card. This device detects the temperature of air, a subtle effect that accompanies breathing. The system consists of a temperature-sensing probe (based on a Siemens silicon-spreading resistance temperature sensor), which detects when the air temperature around it decreases below 0.3°C. We used a comparative sensing technique to change current changes into gradient data. This device detects the changes in temperature that occur when a person is breathing. To avoid calibration, an additional temperature-sensing component placed in a box is used for reference. Then, by comparing the two sensors we can determine when warm air is flowing in front of the user, a sign of the user's breathing. This temperature differential is considered to be a sound signal and is digitized using the sound card. We use the program 'fmod' to control processing (see Figure 1).

The system was reimplemented for 'Crowds'. We now can specify the use of the sensors by additional XML tags, which makes it much easier for the artist. Although there are not many changes from the user's point of view, one important change is that the new system does not use lead wire as the initial system did. This makes it safer for us to use.

In 'Fluids', effects were also hardwired in the C code. This was also changed for 'Crowds'. The system now uses special tags that allow the use of shader effects (implemented in OpenGL shading language) such as 'glow' and 'bump mapping'.

In fact, 'Crowds' involved a major reimplementation and rewrite of the 'Fluids' system that has made it easier for the artist to directly define the scenes by specifying images and XML scripts and importing objects, characters and shaders.

2.3 *Crowds: the installation*

'Crowds' is an interactive installation that requires stereo projection. Stereo can be produced through passive or active devices, depending on the specific setup, and the change only requires

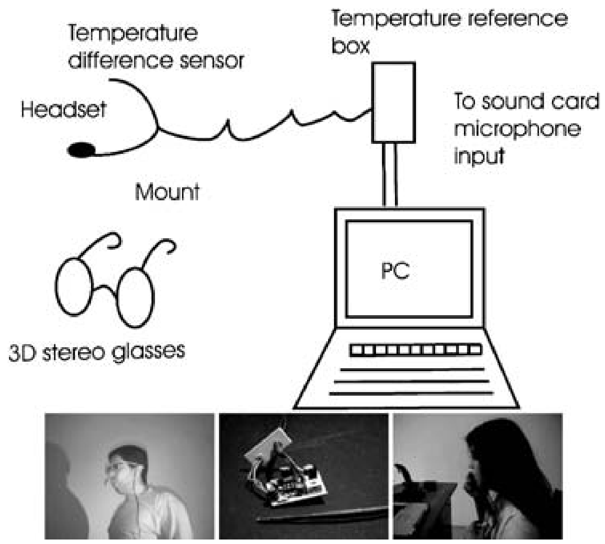

Figure 1 Breath interface and device

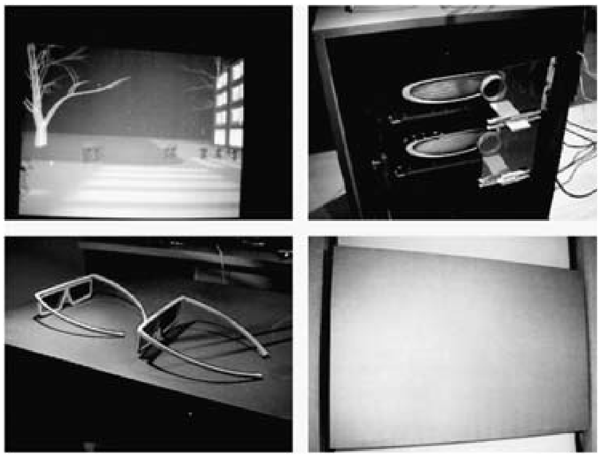

Figure 2 Installation using passive stereo

changing a few parameters in the XML script, which include the type of stereo vision that will be used, and the eye separation between both cameras. In Figure 2 we see the passive stereo setup implemented for current installation.

As an art piece, we felt that 'Fluids' was a very ambitious installation, but we are having the user explore novel navigation, air, and water interfaces, all at the same time, was interesting, but difficult for the user and overly complex for the artist. For 'Crowds' we decided to focus on wind sensing. Joysticks were chosen for navigation. We developed three different scenes. The air temperature sensors described above were tied to the speed of falling snow in all three scenes. This allows both the artist and the users to explore this interface in depth.

'Crowds' explores virtual spaces, which unlike those in 'Fluids', are populated by many characters. Quake 2 characters were used because they are familiar to users and allowed us to convey a specific cartoony feeling we were seeking. In these environments, the characters are engaged in different activities. This changes the nature of the piece from a tour into a simple narrative.

The piece consists of the following scenes:

1. A long lost city of which only a single building is left. The building is visited by strange, shady characters that line up in the entrance, as if waiting for something, perhaps a party, to start.

In this scene the snow is controlled by the user's breathing patterns. The whole scene is rendered in sepia, alluding to black and white films and antique photography (see Figure 3).

2. A futuristic, fantastic city, similar to the city that existed in the past, but now populated by many characters, going nowhere in a hurry. The colors in this city can be changed by the breathing patterns of the user (see Figure 4).
3. Inside the building of the first scene a wild party of varied characters is taking place (see Figure 5). The building is now rendered more realistically, using effects such as bump mapping, but the characters seem to glow, giving the whole scene a surreal effect.

Figure 3 Sepia building

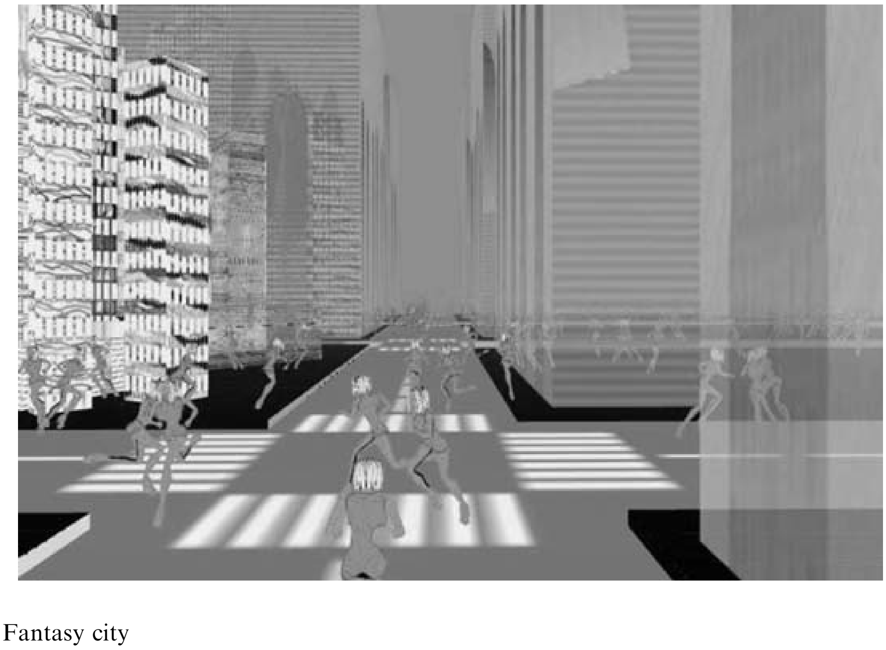

Figure 4 Fantasy city

Figure 5 Party

The scenes described here were helpful in testing out our system. From the point of view of the user, the experience of populated spaces was much more interesting than that of the empty spaces in 'Fluids'. Users lingered much longer. Compared with 'Fluids' interaction worked better because users could concentrate on only one new device, while navigation was simple and traditional. The use of images and scripts allowed the artist to set up the scenes in a very simple fashion. Scenes, including buildings, characters, effects, transitions and even interfaces, could be changed without recompiling, making the system much more artist-friendly than the system we had used for 'Fluids'.

3 UnderCrowds

In large cities there is a behavior of social indifference. Most of the time, when a crowd congregates it is watched by video cameras and other security devices. In mass transit systems, entry and exit are registered. 'UnderCrowds' approaches the esthetics of massive social indifference. It shows the crowds, as detected by security technology, with new forms of visual representation. Now the individual can identify the innate behavior of its species within the different cultural systems in the world. Crowds living in different cities can connect beyond the security systems, in a place where there is nothing that can be censored. Today 75% of the human population live in cities, making crowds into phenomena better observed today than in the past.

At the beginning of the twentieth century, esthetics representing crowds arose, in particular in Russian constructivism. Photography, film, posters, architecture, design, literature, to mention some disciplines, gave legitimacy to crowds as a subject. The esthetic impact was deemed proportional to the scale of the crowd, and still the increase in size of a crowd is correlated with awe. The 'desire of the majorities' is feared, because it is in direct conflict with the idea of individuality.

This installation thus was developed to require scenes that define and simulate large crowds. We developed a system whereby several hundred thousand characters can be simulated and rendered in real-time by using the GPU for both simulation and rendering. 'UnderCrowds' is an installation projecting crowds captured by acoustic and optical sensors. In the screen, one can see the number of individuals the device has registered. The graphical power of our system reaches 260 thousand 3D characters in real-time, simulating human behavior, using a single graphics card.

In contrast to the police race to capture crowds by using video camera systems in cities, 'UnderCrowds' has a naturalistic accent: it is interested in the behavior of crowds, namely stampedes, leader following, appearance and disappearance of leadership, street routine, indecision, etc.

In this work, there is a parameter of conceptual cleanliness. The crowds here presented are not of interest to political or religious leaders, because they only show the gregarious condition of humans. They do not talk among themselves, although collision detection has been implemented. We could say that they 'respect each other electronically. This respect is for proxemic rules, a cultural lock that evokes the indifference of crowds in cities.

Masses are entities erasing the idea of individuals. When the crowd is the repetition of the same character, it alludes to the dictatorship of individuality in contraposition to the blind dictatorship of the majorities.

In this work, simulation of crowds is not a social event and does not represent a particular society, but will work for all of them if regional rules of behavior are taken into account. One could think of it like building a herd of humans: with the same monotony of herds of buffalos, zebras or other animals we have seen in documentaries. 'UnderCrowds' reports the masses as conceptual art. Here the crowds have autonomy to make their own decisions, decide their behavior, recognize themselves as a number, as data. In the current state of research in real-time rendering of crowds, characters inhabiting the virtual world cover the necessities of space and data. Acceding to crowds means observing their behavior, with all the regime of indifference that characterizes them, even when a real spectator sees him/herself reflected in the system.

In the frontier of this research, the oniric condition is raw material for the oniric possibility, for the construction of senses beyond the crowds in movies, political propaganda or nostalgia in the graphics arts.

3.1 Simulation and rendering of Large crowds on the graphics processor

As we added more and more characters, the 'Crowds' system started slowing down significantly. That is why we had to change our simulation system for 'UnderCrowds'. Computational power in graphics processor, or GPUs for short, has dramatically increased over the past decade. Taking also into account that these new GPUs incorporate features such as programmability and high-precision calculations at a low economic cost, they have evolved into extremely flexible and powerful general-purpose processors.

We developed a technique suitable for large crowds of characters that takes advantage of existing programmable graphics hardware (Rudomin *et al*., 2005b). Simulation of large numbers of characters in real-time is made possible by a GPU behavior simulation that uses finite state automata implemented on the GPU. Behavior is specified as an FSM, which is stored as an image on the GPU memory. Additional world and agent maps are used to provide information on the environment and on the state for each agent.

By using a custom shader program, the behavior of each individual agent is simulated in real-time. This shader program queries both agent maps and world maps to get information on the environment. Then, using this information, the FSM map is queried to update the state and position of each agent. In addition, a weight map is produced by the shader, which will specify how to combine world map information according to the agent state. As GPUs are inherently parallel processors, simulations of several agents are executed simultaneously, increasing the frame rate of the application.

One of the main bottlenecks when using the GPU is the data transfer between the graphics memory and the main memory. In order to reduce this data transfer, most parts of the simulation and rendering of crowds are performed within the GPU. We combine two different approaches so as to enable the display of large crowds of animated characters: impostors are used for low-detailed, distant characters, whereas instancing is used for detailed, nearby characters (Millan & Rudomin, 2006).

Impostors are an image-based approach, where each character is replaced by a texture quad resembling as close as possible the appearance of a 3D model from the current point of view. Since

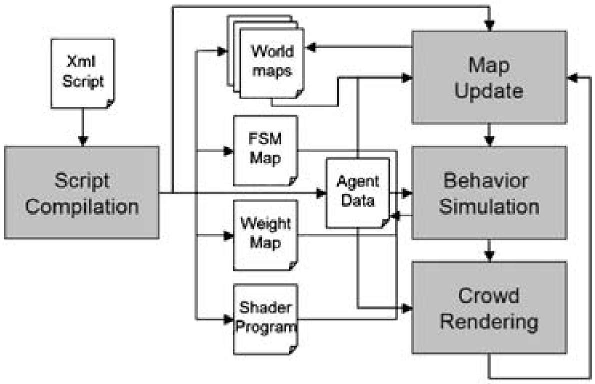

Figure 6 Simulation of large crowd behavior on the GPU based on an XML script

the geometry is reduced to one or two polygons for each character, this technique allows a very efficient rendering of crowds of over a hundred thousand characters.

On the other hand, instancing is a technique that speeds up the rendering of similar geometries by uploading vertex information of such geometries to the graphics memory. This representation is then modified according to a small set of parameters, such as color, texture or transformations, to produce a different looking instance of such an object.

Distance to the camera to each character must be estimated to select the appropriate detail. This is achieved by using a LOD map, where the viewing frustum is projected into the scene plane, producing a shaded polygon where areas close to the camera are brighter. To obtain the detail for each character, this map is queried, obtaining the distance of each character to the camera. Furthermore, as areas outside the frustum are black, it is possible to decide, based on this map, whether a character is visible or not, and cull those characters that do not need to be drawn.

In order to simplify behavior definition for the artist, an automatic translator of an XML script was programmed. This translator reads a script and a set of world maps, and produces the FSM map that describes the scripted behavior, as well as a custom shader program that will evaluate this FSM map and the input world maps to simulate the behavior of large crowds of characters.

A general diagram of the simulation process is shown in Figure 6. Starting from the XML script, and from the set of world maps specified in such a script, the FSM map and the weight map are created, as well as the agent data that will contain the position and state of all characters. A custom shader program is created; this shader changes according to the maps actually used, and will have methods to query such maps in order to select a proper transition from the FSM and to produce a displacement for characters.

After creating and loading these maps, the simulation process starts by updating those maps that change dynamically through time according to character information. After this update, the FSM shader simulates the behavior of crowds, querying the updated maps for world information. Finally, information on agents is used to render the scene and the characters from the crowd currently visible.

3.2 Counting crowds

It is impossible to accurately know how a crowd is behaving and how many actors are involved in this behavior. However, in order to relate a numerical value to human activity, we designed a system to capture and analyze crowds by sensing and counting people.

The human body generates heat and therefore radiation in the infrared spectrum, which can be detected and separated from that generated by other heat-emitting objects. To detect a moving body, one can sense the changes in temperature in the environment. This can be achieved by checking the distortions in the spectrum below 700 nm.

Figure 7 Crowd counter

Using temperature sensors to measure human activity in an open area where lots of people gather is rather complex and there is a need to narrow the sensor's field of view to sense only a specific group of people in one reading.

A sensor detects an obstruction when an infrared ray is obstructed by an opaque object. At this wavelength this can be sensed easily with emitters in the range of 930 nm, but it is a technology that cannot delimit movement or be adjusted to detect only passing humans.

Usually counters modify the behavior of the crowd to do their job and use motion sensing instead of heat sensing to register each user, but in our application we should not interfere with crowd behavior, requiring our crowd counter to be inconspicuous but effective. Our system also needs to be portable. Therefore, after trying several other techniques, we designed a system that uses long-wave infrared sensors, devices that return a nonlinear voltage proportional to the infrared radiation received by the sensor. One can then directly trigger a counter. The sensor is shown in Figure 7.

3.3 UnderCrowds: the piece

'UnderCrowds' contains a set of scenes that convey the feeling of being part of large crowds of people. Given that photorealism for large crowds is still difficult in real-time simulations, and because of esthetic reasons, in 'UnderCrowds' we chose to use the same comic strip look used for 'Crowds'. The initial comic-style screen serves as a gateway that presents and leads to several crowd simulations. This main screen is shown in Figure 8.

Large crowds of characters demand a large amount of computational resources. Since 'UnderCrowds' consists of several scenes each involving different, potentially large crowds, a set of screens and screen transitions was incorporated into the application using a slideshow analogy to avoid displaying and simulating all crowds all the time. XML tags were used to control these screen transitions.

Using this slideshow analogy, screens were handled as slides. Each screen was capable of displaying and simulating a different number of scenes, where each scene portrays a separate crowd simulation. While all crowd simulations are initialized with the application, not all of them are simulated all the time. Instead, only those scenes shown are simulated and updated.

Continuing with the slideshow analogy, screens were changed using screen transitions. These screen transitions control the flow of the application. There are different ways of triggering these transitions, such as pressing a key on the keyboard, waiting for a certain time, clicking on a certain area of the screen, etc. Some transition effects were incorporated to enhance the appearance of such transitions.

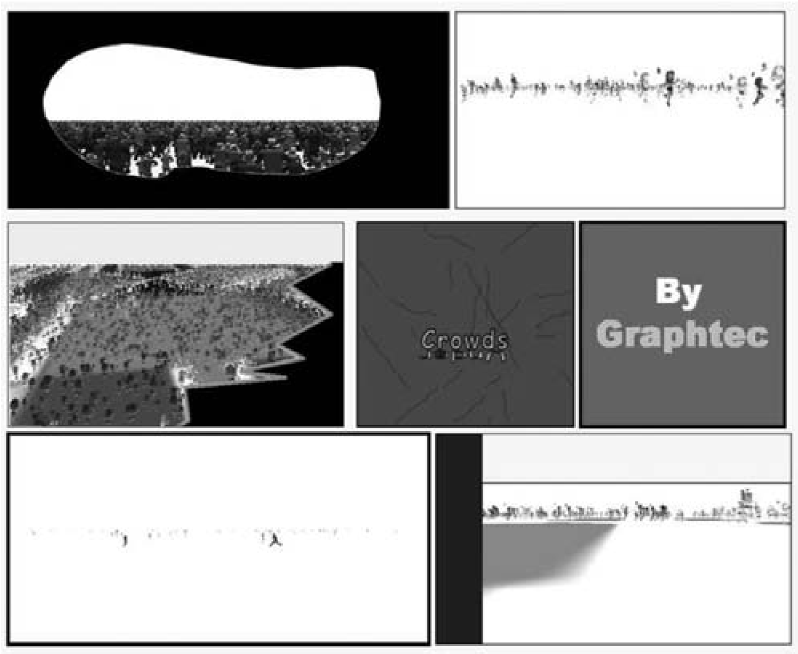

Figure 8 Under Crowds interface

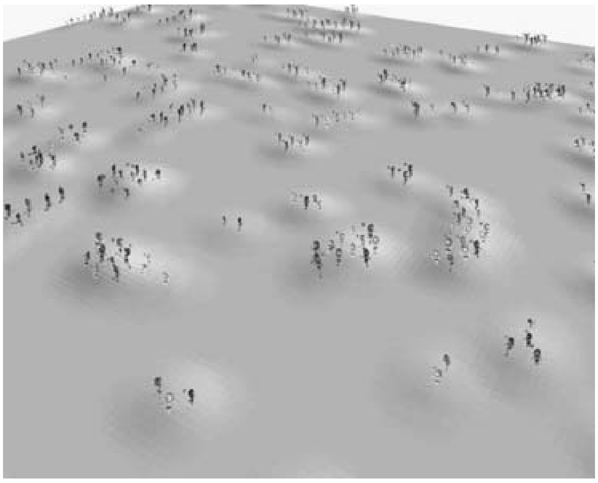

Figure 9 Wandering crowd included in UnderCrowds

From the initial screen, it is possible to navigate towards the different crowd scenes, which include the following. We describe each scene, and also some auxiliary maps that allow the behaviors in the scene to happen. Note that each scene exemplifies different crowd behaviors.

1. Crowds of people wandering around, moving in circles and grouping together. A sample image of this scene is shown in Figure 9. This picture also shows the underlying direction map, which is constantly updated by each character. In this map, each character paints a circle of proximity. Since all characters paint into the same map, more intense patterns are created where more characters are gathered. A similar map is used to avoid collisions between characters, and to make characters move in circles. All these maps are used by a script to generate the desired behavior.
2. In the second scene, we use the crowd counter to monitor the number of people attending the installation, and to modify the number of characters in the virtual crowd to match this number.

Each person attending the installation, and passing in front of the sensor, adds a character to the virtual crowd. Figure 10 shows the resulting scene, where these crowds, in numbers proportional to the counter, are following a small set of leaders. A set of maps similar to those used in the wandering scene described previously was used to make characters follow the leaders (leaders are just characters with very big circles of proximity).

3. A third scene consists of people gathering for a public demonstration. A crowd of characters are attracted towards an area of interest, avoiding different barriers (or maybe barricades) that are found in the environment. Figure 11 shows a snapshot of this scene once an important crowd has been concentrated around the demonstration spot. Here, different influence maps are combined into a single behavior: characters avoid each other when they are close, avoid barriers when close to them and head towards the demonstration site. In this scene, it is interesting to notice how the demonstration area becomes more and more crowded as more people arrive to the site; newly arriving characters push those that have arrived earlier, reducing the separation between them, as it happens with real-life crowds.
4. Figure 12 shows a scene where crowds blend together to create different flags of the world. This creates an interesting effect, where the color for each character changes over time, and when seen from far away, it is hard to distinguish individual characters but rather a whole crowd.

Figure 10 Scene from UnderCrowds, where people detected by the crowd counter create a crowd that follows a small set of leaders

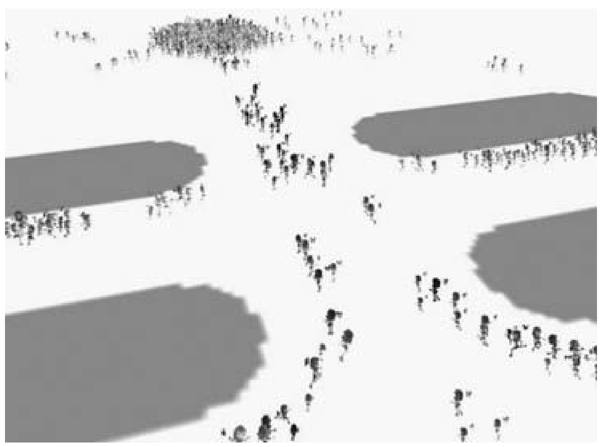

Figure 11 Crowd gathering for a public demonstration in UnderCrowds

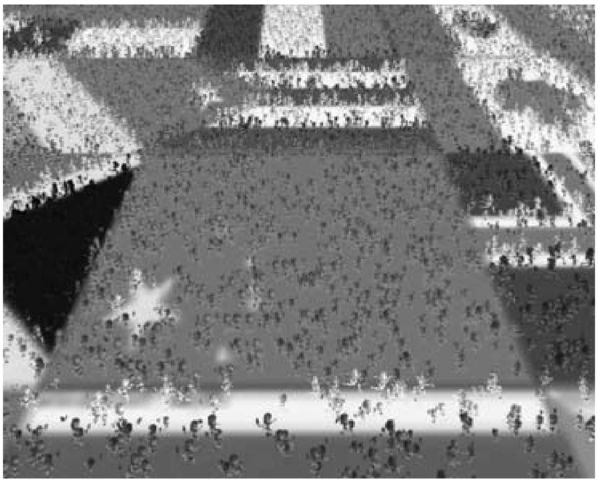

Figure 12 Crowds blending together over different flags in UnderCrowds

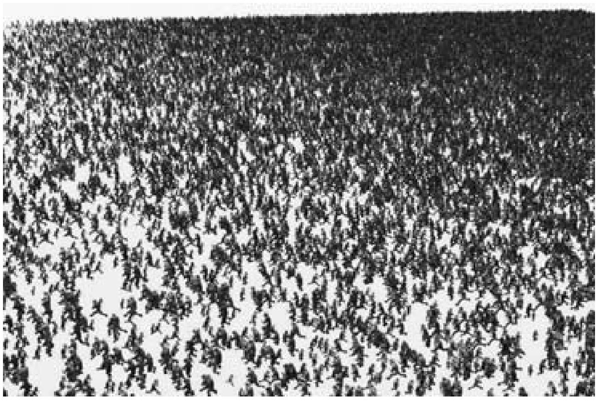

Figure 13 Large crowd of characters included in UnderCrowds

5. The final scene, shown in Figure 13, constitutes a massive crowd of over 50 000 characters, and attempts to convey the feeling of being trapped within a very large crowd. In this scene we use the potential of our system to display large numbers of characters. However, if we add more characters, which can be done easily and can still work in real-time with up to 250 000 characters, the effect is of such overcrowding, which makes it difficult to observe the behavior of the crowd.

4 Conclusions

We developed two different systems, one for each art installation. 'Crowds' is based on the system built for 'Fluids', using some of the sensors originally developed for 'Fluids'. However, the system is much modular and reusable now. The system allows the artist to construct scenes that are populated by different characters and where interaction with several novel devices can take place. The artist can specify simple narratives within the scenes by writing a script describing the activities of the characters within the scene. The visual artist must work more like a film director, but retaining the interactivity of the scenes. This is a vast unexplored territory, and 'Crowds' is only a first, timid attempt to make sense of what this new medium of populated mixed reality will mean.

'UnderCrowds' is a major reimplementation of the system used in 'Crowds'. On the technical aspect it improved the efficiency by using culling and level of detail techniques as well as using the GPU for the actual simulation of the behavior of the characters. Behavior definition uses maps and scripts in a manner similar to that used for 'Crowds', but in this case this specification is translated to custom shaders that are executed within the GPU.

In both pieces we have worked with 'Quake 2' characters because they are of low polygonal complexity, and because they conveyed the look we were seeking. In the future we will be working with more realistic models. We are also working on improving the interface devices we have used and developing others.

References

Cremer, J., Kearney, J. & Papelis, Y. E. 1995. HCSM: a framework for behavior and scenario in virtual environments. *Modeling and Computer Simulation* **5**(3), 242–267.

Davies, C. 2003. Landscape, earth, body, being, space and time in the immersive virtual environments Osmose and Ephemere. In *Women, Art, and Technology*, Malloy, J. (ed.). The MIT Press, 322–337.

Díaz, M. & Rudomin, I. 2005. Designing interfaces for art applications. In *Proceedings of the International Conference on CYBERWORLDS*.

Gebhard, P., Kipp, M., Klesen, M. & Rist, T. 2003. Authoring scenes for adaptive, interactive performances. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems*. ACM Press, 725–732.

Grau, O. 2003. *Virtual Art: From Illusion to Immersion*. MIT Press.

Ishii, H. 1999. Tangible interfaces. In *Proceedings of the Siggraph 2005*. ACM Press, 127.

Millán, E. & Rudomin, I. 2006. Impostors and pseudo-instancing for gpu crowd rendering. In *GRAPHITE '06: Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*. ACM Press, 49–55.

Musse, S. R. & Thalmann, D. 2001. Hierarchical model for real time simulation of virtual human crowds. *IEEE Transactions on Visualization and Computer Graphics* **7**(2), 152–164.

Reynolds, C. W. 1987. Flocks, herds, and schools: a distributed behavioral model. *Computer Graphics* **21**(4), 25–34.

Rudomin, I. & Millán, E. 2004. XML scripting and images for specifying behavior of virtual characters and crowds. In *Proceedings of the CASA 2004*. University Press, 121–128.

Rudomin, I., Perez, F. & Millán, E. 2004. Groups and crowds with behaviors specified in the environment. In *Proceedings of the IVEVA*.

Rudomin, I., Díaz, M., Hernandez, B. & Rivera, D. 2005a. Water, temperature and proximity sensing for a mixed reality art installation. In *Proceedings of the INtelligent TEchnologies for Interactive Entertainment (INTETAIN)*.

Rudomin, I., Hernandez, B. & Millán, E. 2005b. Fragment shaders for agent animation using finite state machines. *Simulation Modelling Practice and Theory Journal Elsevier (preprint)* **13**(8), 741–751.

Rudomin, I., Hernandez, B. & Millán, E. 2005c. Probabilistic, layered and hierarchical animated agents using XML. In *Proceedings of the Graphite*.