Challenges in Crowd Simulation

Daniel Thalmann, Helena Grillon, Jonathan Maim, Barbara Yersin
EPFL VRlab
Contact: Daniel.Thalmann@epfl.ch

Abstract

The purpose of this paper is to identify the problems to solve in order to simulate real-time crowds in a Virtual Environment. We try to classify these problems and study how they have been addressed until now by the research community and our Lab in particular. We then discuss for each problem what are the future challenges and how to address them.

Keywords: Crowd simulation, behavioral animation, gaze, variety

1. Introduction

The long-term objective of our research since a decade is the real-time simulation of virtual crowds evolving in different environments. We have achieved most of our general objectives. In the past, work on crowds at VRlab has focused on rendering realistic real-time massive amount of virtual characters. The current crowd engine rendered virtual characters that navigate onto an environment-embedded graph. It has a scalable architecture to support tens of thousands of characters. A lot of attention was dedicated to generate uniqueness within the crowd. From a set of few character templates and parameters, the engine creates variety among virtual humans. A character is unique because of its template model, its texture, the colors of its body parts, and all the accessories it wears.

Before identifying the problems and the challenges, it is essential to fix our criteria for real-time crowd simulation. "Real-time" is a first criterion, this means that we want to have the simulation running at a frame rate of 30 Hz. Also, we have to clarify what is a crowd; although there is no official minimum requirement for a definition of crowd, we consider that a crowd should have more than one thousand individuals to be qualified of crowd. We also consider as an important criterion the realistic aspects. A crowd should have a nice appearance which means nice virtual humans but also we expect that individuals are mostly different. This means of appearance, shape, height, clothes, but also in terms of animation and behavior. Concerning the motion, we expect fluid motion and collision avoidance.

Our goal is to simulate crowd behavior in specific situations, reacting to events, having a goal. This means that we need to create high-level behaviors. In particular, we should have a flexible path planning and of course to generate spontaneous movements and adaptive movements. It is also important that the user can be part of the crowd. This means that the crowds should be a world of agents while the user is represented by an avatar. But, it is essential that the avatar looks similar to the rest of the crowd.

2. Variety

To generate thousands of individuals, a naive approach is to design as many humans as there are people in the crowd. Obviously, such an approach is impossible, since it would require armies of designers, and an infinite memory. The common and more reasonable approach is to use human templates. A human template is a virtual human defined by its skeleton, its mesh, which is skinned to the skeleton, and its set of textures. To create large crowds, a small group of human templates are instantiated several times. For each instance, one texture is randomly chosen within the template's available set. Then, color and shape variety techniques are applied, so that instances of a same template and using the same texture are still different [1]. To generate many locomotion animations for each human template, we use a motion-capture-based locomotion engine [2,3]. This allows us to obtain adapted locomotion cycles that will be used at runtime to animate the characters. Using an IK system, upper body variations are also applied to further vary the crowd: the characters are thus able to have a hand in their pocket, or next to their ear to make a phone call. Note that the generated animations are skeletal, i.e., they can be applied to meshes that are deformed at runtime. In our crowd

engine, we use such meshes only at the forefront of the camera, for the animation process is very costly. The second and third levels of detail we use, i.e., static meshes and impostors (see Section 8.1), are pre-computed and saved in the database too.

3. Appearance variety

3.1. Color variety

Previous work on color variety is based on the idea of dividing a human template into several body parts, identified by specific intensities in the alpha channel of the template texture. At runtime, each body part of each character is assigned a color in order to modulate the texture. Tecchia et al. [4] used several passes to render each impostor body part. Dobbyn et al. [5] extended the method to 3D meshes and avoided multi-pass rendering with programmable graphics hardware. Although these methods offer nice results from a reasonable distance, they produce sharp transitions between body parts. Based on the same idea, Gosselin et al. [6] showed how to vary characters with the same texture by changing their tinting. They also presented a method to selectively add decals to the characters' uniforms. However, their approach is only applied to armies of similar characters, and the introduced differences are not sufficient when working with crowds of civilians.

Using a single alpha layer to segment body parts has several drawbacks. No bilinear filtering can be used on the texture, because incorrect interpolated values would be fetched in the alpha channel at body part borders. Moreover, for individuals close to the camera, the method tends to produce too sharp transitions between body parts, e.g., between skin and hair, due to the impossibility of associating a texel to several body parts at the same time. Also, character close-ups bring the need for a new method capable of handling detailed color variety. Subtle make-up, or detailed patterns on clothes greatly increase the variety of a single human template. Furthermore, changing illumination parameters of materials, e.g., their specularity, provides more realistic results. Previous methods would require costly fragment shader branching to achieve such effects. We apply a versatile solution based on segmentation maps to overcome previous method drawbacks.

For each texture of a human template, we create a series of segmentation maps. Each of them is an RGBA image, delimiting four body parts, i.e., one per channel, and sharing the same parameterization as the human template texture. This method allows for each texel to partially belong to several body parts at the same time through its channel intensities. As a result, it is possible to design transitions between body parts much smoother than in previous approaches..

3.2. Shape variety

Magnenat-Thalmann et al [7] classified the methodologies for modeling virtual people into three major categories: creative, reconstructive, and interpolated. Geometric models created by artists such as anatomically based models fall into the former approach. The second major category built 3D virtual human's geometry by capturing existing shape from 3D scanners, images and even video sequences. The interpolated modeling uses sets of example models with an interpolation scheme to reconstruct new geometric models. For crowds, we can surely forget the first approach which is too expensive in terms of manual work. The second way is also prohibitive for large crowds and also present a lack of flexibility. The last approach is the most convenient. We can cite two examples of such an approach. For large crowds, another approach consists in modifying separately the height of the human body and its shape.

3.2.1. Height. We can modify the height of a human template, by scaling its skeleton. To help the designer in this task, we provide additional functionalities to the design tool presented above: for a given human template skeleton, the global space of height scaling can be defined. Fine-grained local tuning for each joint can also be specified, i.e., minimal and maximal scale parameters on the x, y, and z world axes. These data allow several different skeletons to be generated from a single template, which we call the meta-skeleton. Figure 1 shows an example.

Figure 1. Variation of height

For each new skeleton, a global scale factor is randomly chosen within the given range. Then, the associated new scale for each of its bones is deduced .

Short / tall skeletons mixed with broad / narrow shoulders are thus created. The skin of the various skeletons also needs adaptation. Each vertex of the original template is displaced by each joint that influences it.

3.2.2. Shape. The human mesh is modified using 3 steps:

Step 1. Using a commercial 3D package like 3DSMax, it is possible for a designer to paint a FatMap for a given template, as seen in Figure 2. The FatMap is an extra gray-scale UV texture that is used to emphasize body areas that store fat. Darker areas represent regions where the skin will be most deformed, e.g., on the belly, and lighter areas are much less deformed, like the head. When the creation of the FatMap is complete, the grayscale values at each texel are used to automatically infer one value for each vertex of the template's mesh. Each of these values, called a fatWeight, is attached to the vertex as an additional attribute.

Step 2. The next step is to compute in which direction the vertices are moved when scaled. For this, we compute the scaling direction of each vertex as the weighted normal of the bones influencing it.

Step 3. Once the direction of the body scaling is computed for each vertex, the actual scaling can take place. The extent to which we scale the body is defined by a fatScale, randomly chosen within a pre-defined range.

Figure 2. FatMaps designed in 3DSMax. Dark areas represent regions more influenced by fat or muscles modification, while lighter parts are less modified.

3.3 Future Challenges in Appearance Variety

The main challenge would be to generate crowds of people with a real different anatomy. The bodies could have really different shapes, but the most important would be to generate completely different faces. Moreover, instead of textures, autonomous clothes could be used like dresses, shirts, skirts etc… Clothes simulation is another approach to further vary characters [8], but for real-time crowd applications, their animation remains too expensive to be used. Dobbyn et al. [9] recently proposed to pre-simulate cloth mesh deformation and use the resulting animation at runtime on impostors. But, the most sophisticated methods for clothing [10] will be used in the future. In fact, these developments are already possible today, but they are not applicable for large crowds in real-time.

4. Accessories

4.1. Modeling Accessories

Accessorizing crowds offers a simple and efficient alternative to costly human template modeling. Accessories are small meshes representing elements that can easily be added to the human template original mesh. Their range is considerable, from subtle details, like watches, jewelry, or glasses, to larger items, such as hats, wigs, or backpacks, as illustrated in Figure 3. Distributing accessories to a large crowd of a few human templates varies the shape of each instance, and thus makes it unique. Similarly to deformable meshes, accessories are attached to a skeleton and follow its animation when moving.

Figure 3. Accessories

The first group of accessories does not necessitate any particular modification of the animation clips

played. They simply need to be correctly "placed" on a virtual human. Each accessory can be represented as a simple mesh, independent from any virtual human. First, let us lay the problem for a single character. The issue is to render the accessory at the correct position and orientation, accordingly to the movements of the character. To achieve this, we can "attach" the accessory to a specific joint of the virtual human. Let us take a real example to illustrate our idea: imagine a walking person wearing a hat. Supposing that the hat has the correct size and does not slide, it basically has the same movement as the head of the person as he walks.

The second group of accessories we have identified is the one that requires slight modifications of the animation sequences played, e.g., the hand close to the ear to make a phone call, or a hindered arm sway due to carrying a heavy bag. Concerning the rendering of the accessory, we still keep the idea of attaching it to a specific joint of the virtual human. The additional difficulty is the modification of the animation clips to make the action realistic. We only focus on locomotion animation sequences. There are two options to modify an animation related to an accessory:

- If we want a virtual human to carry a bag for instance, the animation modifications are limited to the arm sway, and maybe a slight bend of the spine to counterweight the bag. Such modifications can be applied procedurally, and at runtime, by blocking some joint movements, and / or clamping their rotation.
- If it is a cellphone accessory that we want to add, we need to keep the hand of the character close to its ear and avoid any collision over the whole locomotion cycle. This kind of modifications are too complex to be achieved at runtime. In such cases, we work with an inverse kinematic tool to modify the animation cycles in a pre-process.

4.2 Future Challenges in Accessories

Firstly, the mesh is presumed to be attached to a single joint, limiting the possibilities. It would be possible to skin an accessory with more joints, but adapting it to any template without changing the mesh vertices would prove to be difficult.

Secondly, to attach accessories to moving characters, we assume to work with skeletons and skeletal animations, which is not necessarily the case. Nevertheless, this limitation can easily be overcome by attaching an accessory, for instance, to one of the vertices composing the character instead of a joint.

Thirdly, the technique does not provide solutions for simulating movements independent from the attach joint, e.g., a too big hat sliding on a child's head.

Finally, some accessories cannot be used as presented here, because their presumed weight should alter the animation of the characters. For instance, a handbag can easily be placed in a virtual human's hand, but if the performed animation sequence is not altered, the bag seems weightless, and the resulting effect is not realistic. Light accessories have little impact on locomotion animation. However holding a bag clearly influences the way arms swing during a walk cycle. Simple solutions have been proposed in past work. However such solutions have limitations that, in some cases, can break the realism of the animation when the camera gets close to the characters.

Currently, individuals in crowds can carry accessories, what we don't see is crowds manipulating objects, open doors, eating, bringing objects from one place to another, exchanging objects.

5. Animation Variety

5.1 How to vary animation ?

A second important factor, although less paramount is their animation. If they all perform the same animation, the results are not realistic enough [11]. In VRlab, we have implemented three techniques to vary the animation of characters, while remaining in the domain of navigating crowds, i.e., working with locomotion animations:

1. We introduced variety in the animation by generating a large amount of locomotion cycles (walking and running), and idle cycles (like standing, talking, sitting, etc, that we morphologically adapt for each template. We take care to make these animations cyclic, and categorize them in the database, according to their type: sitting or standing, talking or listening, etc. For locomotion clips, walk and run cycles are generated from a locomotion engine based on motion capture data (see Section 5.2). We compute such locomotion clips for a set of speeds. Thus, during real-time animation, it is possible to directly obtain an adequate animation for a virtual human, given its current locomotion velocity, and its morphological parameters. Then, we designed the concept of motion kit, a data structure, that efficiently handles animations at all levels of detail (LOD).

2. We designed a second technique of animation variety, i.e., how pre-computed animation cycles can be augmented with upper-body variations, like having a hand on the hip, or in a pocket.
3. Finally, we introduced procedural modifications applied at runtime on locomotion animations to allow crowds to wear complex accessories as mentioned in the previous Section.

5.2 Locomotion Animation Clips

To generate our original set of walk and run cycles, we used the locomotion engine developed by Glardon et al. [2]; it is an integrated walking and running engine able to extrapolate data beyond the space described by the PCA basis. In this approach, the Principal Component Analysis (PCA) method is used to represent the motion capture data in a new, smaller space. As the first PC's (Principal Components) contain the most variance of the data, an original methodology is used to extract essential parameters of a motion. This method decomposes the PCA in a hierarchical structure of sub-PCA spaces. At each level of the hierarchy, an important parameter of a motion is extracted and a related function is elaborated, allowing not only motion interpolation but also extrapolation. Figure 4 shows an example.

There are mainly three high-level parameters which allow to modulate these motions:

- *Personification weights*: several people, different in height and gait have been captured while walking and running. This variable allows the user to choose how he wishes to parametrize these different styles.
- *Speed*: the subjects have been captured at many different speeds. This parameter allows to choose at which velocity the walk/run cycle should be generated.
- *Locomotion weights*: this parameter defines whether the cycle is a walk or a run animation. Thus, the engine is able to generate a whole range of varied locomotion cycles for a given character. Each human template is also assigned a particular personification weight so that it has its own gait. With such a high number of animations, we are already able to perceive a sense of variety in the way the crowd is moving. Virtual humans walking together with different locomotion styles and speeds add to the realism of the simulation.

Figure.4. PCA-based walking models

5.3 Future challenges in Animation Variety

The challenges are of different types:

1) The adaptation of animation clips to the various morphologies. This means, for example, adaptation of walking to tall people, to fat people.
2) The realization of animation sequences when collisions with the environment occur. For example, people can hit a wall or even move a table by hitting it.
3) The development of more complex animation clips such as climb stairs or sit down

Finally, the big challenge would be to replace clips by an online motion generator able to produce spontaneous movements according to the situation. This kind of motion generator is extremely hard to produce and cannot handle tens of thousands of virtual humans.

6. Interface for Crowds

When increasing the number of involved individuals it is becoming more difficult to create unique and varied content of scenarios with large numbers of entities. If we want to create or modify features of every single individual one by one, it will soon become too laborious. If, on the other hand, we apply a set of features to many individuals at once, it could create unwanted artefacts on a larger scale, resulting in an "army-like" appearance with too uniform, or periodic distributions of individuals or characteristics. Use of random distributions can alleviate such problems; however, it can be very difficult to capture the desired constraints into a set of mathematical equations, especially considering integration into common art production pipelines.

6.1 Brush Metaphor

We introduced CrowdBrush [12], an innovative approach to create complex scenes involving thousands of animated individuals in a simple and intuitive way. By employing a brush metaphor, analogous to the tools used in image manipulation programs, we can distribute, modify and control crowd members in real-time with immediate visual feedback. Brushes are tools with visual representation on the screen that affect crowd members in different manners: for example, the brush can create new individuals in the scene, or it can change their appearances or behaviors. We selected visualizations of the brushes to intuitively hint on function. For example, the creation brush has an icon of a human, the orientation brush has an icon of a compass, the deletion brush has an icon of a crossed over human, and so on. The brush is processed in three stages. First, a selection of the affected area in the 2D screen space is performed according to a triggered mouse button, with subsequent picking of the entities in the 3D world space. Then, the operator will modify the manner of execution of the brush in the selected area. Finally, the brush will change the values of the instance properties for the affected individuals. In case of the creation brush, it will create new population members; for the event brush it will send events to a behavior system and for the path brush it will add a new waypoint to a current path. Figure 5 shows a brush to create people.

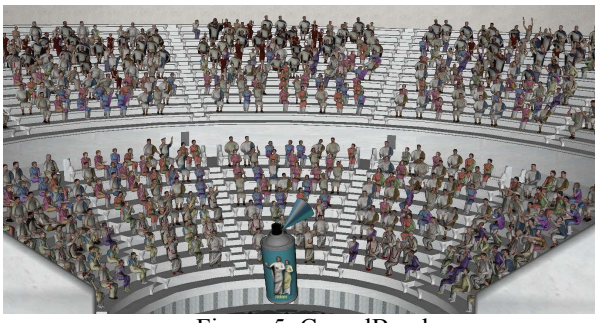

Figure 5. CrowdBrush

6.2 Future Challenges in Interface for Crowds

The main limitation of our spatially oriented brush metaphor is a weak control of time related aspects of scenarios. We address this issue by incorporating behavior rules engine that is complementary to brushes. A more limited possibility is to provide other authoring methods as scripts that can be used in conjunction with the brush metaphor. Finally, it would be also possible to address individuals in a crowd using gestures and speech.

7. Motion Planning

7.1 Navigation Graphs

Real-time crowd motion planning requires fast, realistic methods for path planning as well as obstacle avoidance. The difficulty to find a satisfying trade-off between efficiency and believability is particularly challenging, and prior techniques tend to focus on a single approach. We have presented [13,14] a novel approach to automatically extract a topology from a scene geometry and handle path planning using a navigation graph. Figure 6 shows a crowd moving using navigation graphs. The main advantage of this technique is that it handles uneven and multi-layered terrains. Nevertheless, it does not treat inter-pedestrian collision avoidance. Given an environment geometry, a navigation graph can be computed [15,14]: the vertices of the graph represent circular zones where a pedestrian can walk freely without the risk of colliding with any static object composing the environment. Graph edges represent connections between vertices. In the environment, they are viewed as intersections (or gates) between two circular zones (vertices). From a navigation graph, path requests can be issued from one vertex to another. Using an algorithm based on Dijkstra's, we are able to devise many different paths that join one point of the environment to another one. It is possible to provide the navigation graph with a second model of the environment, which usually has a much more simple geometry, annotated with information. This second model is automatically analyzed, its meta-information retrieved, and associated to the corresponding vertices.

Figure 6. Crowd moving

7.2 Scalable Motion Planning

More recently, Treuille et al. [16] proposed realistic motion planning for crowds. Their method produces a potential field that provides, for each pedestrian, the next suitable position in space (a waypoint) to avoid all obstacles. Compared to agent-based approaches, these techniques allow to simulate thousands of pedestrians in real time, and are also able to show emergent behaviors. However, they produced less believable results, because they require assumptions that prevent treating each pedestrian with individual characteristics. For instance, only a limited number of goals can be defined and assigned to groups of pedestrians. The resulting performance depends on the size of the grid cells and the number of groups

7.3 An hybrid Architecture based on Regions Of Interest (ROI)

We proposed [17] a hybrid architecture to handle the path planning of thousands of pedestrians in real time, while ensuring dynamic collision avoidance. The scalability of our approach allows to interactively create and distribute regions of varied interest, where motion planning is ruled by different algorithms. Practically, regions of high interest are governed by a long-term potential field-based approach, while other zones exploit a graph of the environment and short-term avoidance techniques. Our method also ensures pedestrian motion continuity when switching between motion planning algorithms. Tests and comparisons show that our architecture is able to realistically plan motion for many groups of characters, for a total of several thousands of people in real time, and in varied environments.

The goal of our architecture is to handle thousands of pedestrians in real time. We thus exploit the above mentioned vertex structure to divide the environment into regions ruled by different motion planning techniques. Regions of interest (ROI) can be defined in any number and anywhere in the walkable space with high-level parameters, modifiable at runtime.

By defining three different ROI, we obtain a simple and flexible architecture for realistic results: ROI 0 is composed of vertices of high interest, ROI 1 regroups vertices of low interest, and ROI 2 contains all other vertices, of no interest.

For regions of no interest (ROI 2), path planning is ruled by the navigation graph. Pedestrians are linearly steered to the list of waypoints on their path edges. To use the minimal computation resources, obstacle avoidance is not handled.

Path planning in regions of low interest (ROI 1) is also ruled by the navigation graph. To steer pedestrians to their waypoints, an approach similar to Reynolds' is used [18], and obstacles are avoided with an agent-based short-term algorithm. Although agent-based, this algorithm works at low level, and thus stays simple and efficient.

In the regions of high interest (ROI 0), path planning and obstacle avoidance are both ruled by a potential field-based algorithm, similarly to Treuille et al. [16].

Figure 7 shows a crowd moving using the hybrid path planning algorithm.

Figure 7. Crowd using hybrid path planning

7.4 Future Challenges in Motion Planning

There are some limitations to our architecture. Firstly, in too crowded narrow environments, severe bottlenecks may appear, making the use of our potential field-based approach a waste of computational time. However, it is possible to enforce a low level of interest in these regions, e.g., ruled by a short-term avoidance algorithm. Another limitation is our group-based approach: we are constrained to assign general goals for groups of pedestrians. One goal per pedestrian would be too prohibitive for real-time applications.

An interesting lead for future work is to merge other avoidance algorithms in the architecture to obtain better results in crowded situations where the space is narrow.

Algorithms simulating each pedestrian individually, with respective characteristics, should be investigated and possibly merged with our ROI architecture. Also, techniques to make the pedestrian animation perfectly synchronized with its motion should be explored. Another aspect of crowds that arouses our interest is the simulation of small groups of people. Indeed, in real life, it is rare to observe people walking alone, and

flocking behaviors are necessary to obtain realistic results.

An interesting lead for future work is to merge other avoidance algorithms in the architecture to obtain better results in crowded situations where the space is narrow. Algorithms simulating each pedestrian individually, with respective characteristics, should be investigated and possibly merged with our ROI architecture. Also, techniques to make the pedestrian animation perfectly synchronized with its motion should be explored. Another aspect of crowds that arouses our interest is the simulation of small groups of people. Indeed, in real life, it is rare to observe people walking alone, and flocking behaviors are necessary to obtain realistic results.

8. Crowd Behavior

8.1 Group Behavior

The behaviour of people in a crowd is a fascinating subject: crowds can be very calm but also rise to frenzy, they can lead to joy but also to sorrow. It is quite a common idea that people not only behave differently in crowd situations, but that they undergo some temporary personality change when they form part of a crowd. Most writers in the field of mass- or crowd- psychology agree that the most discriminating property of crowd situations is that normal cultural rules, norms and organisation forms cease to be applicable For instance in a panic situation the normal rule of waiting for your turn, and the concomitant organisation form of the queue, are violated and thus become obsolete.

In Musse et al [19], the model presents a simple method for describing the crowd behavior through the group interrelationships. Virtual actors only react in the presence of others, e.g., they meet another virtual human, evaluate their own emotional parameters with those of the other one and, if they are similar, they may walk together. The group parameters are specified by defining the goals (specific positions which each group must reach), number of autonomous virtual humans in the group and the level of dominance from each group. This is followed by the creation of virtual humans based on the groups' behavior information. The individual parameters are: a list of goals and individual interests for these goals (originated from the group goals), an emotional status (randomic number), the level of relationship with the other groups (based on the emotional status of the agents from a same group) and the level of dominance (which follows the group trend). The sociological effects modeled in the presented rules are:

- *grouping* of individuals depending on their inter-relationships and the *domination* effect;
- *polarization* and the *sharing* effects as the influence of the emotional status and domination parameters; and finally,
- *adding* in the relationship between autonomous virtual humans and groups.

The group behavior is formed by two behaviors: seek goal, that is the ability of each group to follow the direction of motion specified in its goals, e.g. in the case of a visit to a museum, the agents walk in the sense of its goals; and the flocking (ability to walk together), has been considered as a consequence of the group movement based on the specific goals during a specific time.

Generally, the available computational resources to trigger intelligent behaviors are very limited in crowds, for their navigation, animation, and rendering are already very expensive tasks that are absolutely paramount. Our approach to this problem is to find a trade-off that simulates intelligent behaviors, while remaining computationally cheap. In [13], we detail the various experiments we have achieved to improve pedestrians behaviors. To make crowd movements more realistic, a first important step is to identify the main places where many people tend to go, i.e., places where there is a lot of pedestrian traffic. It can be a shopping mall, a park, a circus, etc. Adding meta-information to key places in an environment has been achieved in many different ways, as introduced in Section 6. Our approach is to use the Navigation Graph of an environment to hold this meta-information, which is a very advantageous solution: instead of tagging the meshes of an environment, or creating a new dedicated informational structure, we directly work on the structure that is already present, and which is used for path planning and pedestrian steering.

8.2 Gaze

We can improve the realism of a crowd simulation by allowing its pedestrians to be aware of their environment and of the other characters present in this environment. They can even seem to be aware of a user interacting with this environment. We introduced [20] the various setups which allow for crowd characters to gaze at environment objects, other characters or even a user. Finally, we developed a method to add these attentional behaviors in order for crowd characters to seem more individual.

The first step is to define the interest points, i.e. the points in space which we consider interesting and

which therefore attract the characters' attention. We use several different methods to do this depending on the result we want to obtain:

- The interest points can be defined as regions in space which have been described as interesting. In this case, they will be static.
- They can be defined as characters evolving in space. All characters may then potentially attract the attention of other characters as long as they are in their field of view. In this case, we have dynamic constraints, since the characters move around.
- They can be defined as a user if we track a user interacting with the system. A coupled head- and eye-tracking setup allows us to define the position of the user in the 3D space. Characters may then look at the user.

The second step to obtain the desired attentional behaviors consists in computing the displacement map which allows for the current character posture to achieve the gaze posture, i.e. to satisfy the gaze constraints. Once the displacement map has been computed, it is dispatched to the various joints composing the eyes, head, and spine in order for each to contribute to the final posture. Finally, this displacement is propagated in time in order for the looking or looking away motions to be smooth, natural, and human-like. Figure 8 shows virtual humans with gaze.

Figure 8. An example depicting the types of possible gaze behaviors

8.3 Future Challenges in Crowd Behaviors

The challenges in crowds behaviors are unlimited. We are only able to generate very simple behaviors. There is a lack of real interaction between people, we need intra-group interactions and extra-group interaction. The social behavior is almost inexistent. Emotions are still very primitive in Virtual Humans and collective emotions are not present in current crowd technology.

9. Beyond the Limits: How to Improve the Performances

9.1 Rendering Strategies

Several strategies and techniques for efficient **rendering of crowds** have been proposed as surveyed in [8]. Many solutions exploit the use of *impostors* to achieve high-performance rendering. Impostors [21] are sets of 2D pre-computed images for virtual humans which are used in place of 3D models when rendering the whole scene. Images are pre-computed using various view angles in order to fit any relative position between camera and humans. More recently, Rudomin and Millan [22] have extensively used impostors for all instances, near or far ones, whereas [5] proposed to add static meshes in the vicinity of the camera and only using impostors at a farther distance.

Our approach [15] also benefits from a level-of-detail strategy: at the fore-front, highly detailed dynamic meshes capable of facial animation are used. Then, at a farther distance, precomputed static meshes are displayed, and finally, when instances appear very small, impostors are used.

9.2 Crowd Patches

We break classical crowd simulation limitations on the environment dimensions: instead of pre-computing a global simulation dedicated to the whole environment, we independently pre-compute the simulation of small areas, called crowd patches [23]. To create virtual populations, the crowd patches are interconnected to infinity from the spectator's point of view. We also break limitations on the usual durations of pre-computed motions: by adapting our local simulation technique, we provide periodic trajectories that can be replayed seamlessly and endlessly in loops over time.

Our technique is based on a set of patch templates, having specific constraints on the patch content, e.g., the type of obstacles in the patch, the human trajectories there, etc. A large variety of different patches can be generated out of a same template, and then be assembled according to designers' directives.

Patches can be pre-computed to populate the empty areas of an existing virtual environment, or generated online with the scene model. In the latter case, some of the patches also contain large obstacles such as the buildings of a virtual city.

9.3 Future challenges

The ultimate challenge would be to be able to generate real-time massive crowds without any artifacts like LODs, impostors, and patches. However, this is not possible and according to the development of hardware, it will take many years before it is possible.

But, keeping the crowd patch approach, we may introduce a few improvements.

A limitation of our solution is the need for an identical period for all interconnected patches. Changing the period is easy when dealing with exogenous objects, as we proposed an automatic technique for computing their periodic animation. However, endogenous objects may have hand-designed animation, making the change of period duration more difficult. Patches are created once at a given place: the overall distribution of the population in the environment is thus static in time. As a result, it is not possible to synthesize dynamic events, such as a small demonstration of people going through the environment. An interesting direction for future work would be to allow dynamic changes of patterns and patches. If spectators' point of view remains static for a long period of time, the animation periodicity may be detected. It is possible to make the detection more difficult by generating a variety of patches from identical patterns, instead of a unique instance.

We can also propose two future directions to allow interaction between the users and the virtual population. The first one would be to locally edit locomotion trajectories in order to account for the presence of spectators. However, patterns define strict limit conditions for these trajectories. Managing edition and limit conditions would then need special care. A second solution would be to consider that patches are only used to simulate secondary characters, whereas interactive digital actors would be added to this background population. Interactions between digital actors and secondary characters would also need careful management.

10. Applications

We can conclude this paper with a few applications and the challenges associated to them

10.1 Virtual Heritage

Based on archaeological data, we have presented the different steps of our work to generate the ancient city of Pompeii and populate it with Virtual Romans [24] (see Figure 9). Thanks to the semantic data labeled in the geometry, crowds are able to exhibit particular behaviors relatively to their location in the city. Our results and show that we are able to simulate several thousands virtual characters in the reconstructed city in real-time. The use of a procedural technique for the creation of city models has proven to be very flexible and allows for quick variations and tests not possible with manual editing techniques.

One possible challenging work would be to make the Virtual Romans interact with the model, e.g., opening doors. This would allow to create more intelligent and varied behaviors for crowds.

Figure 9. Roman crowd in Pompeii

10.2 Transportation and urbanism

Virtual crowds are used for simulation of new train stations and airports. A challenge would be to introduce natural motivations to simulate more complex and realistic situations. For example, in an airport, people should not just check in, go to the security then the gate, as in most simulations. They should be able to go to restaurants, cafés, shops, toilets, according to their internal motivations. Such models exist, but the problem is that it will be extremely CPU intensive to introduce them to

10.3 Agoraphobia treatment

We developed an application allowing for characters to perform gazing motions in a real-time virtual crowd in a CAVE environment. Moreover, it allows for users to interact with those crowd characters. It is an adaptation of the model of visual attention described in [20] in order to integrate it in a crowd engine and allow for the method to function online (in real-time). Certain aspects of the automatic interest point detection have been greatly simplified. The existing architecture has been also modified in order to abide with the limitations induced by the realtime implementation.

The final application consists in a city scene, projected in a CAVE setup, and in which a crowd of characters walks around (see Figure 10). The application uses a Phasespace optical motion capture device to evaluate where a user is looking and more specifically, which character he/she is looking at. Finally, we further enhance this setup with an RK-726PCI pupil/corneal reflection tracking device in order to evaluate more precisely where a user is looking. The system then allows for the crowd characters to react to user gaze. For example, since we can determine the user's position and orientation in the virtual world, the characters can look at the user.

Figure 7. Crowd using hybrid path planning

Acknowlegments

Research on crowd has been partially sponsored by the Swiss National Research Foundation and the CyberEmotions EU project.

11. References

[1] J. Maïm, B. Yersin, and D. Thalmann, Unique Instances for Crowds. To Appear in IEEE Computer Graphics and Applications, 2009.

[2] P. Glardon, R. Boulic, and D. Thalmann, PCA-based walking engine using motion capture data. In: Proc. of Computer Graphics International, 2004

[3] P. Glardon, R. Boulic and D. Thalmann, Robust on-line adaptive footplant detection and enforcement for locomotion, Visual Computer, Vol. 22, Nr. 3, pp. 194-209, 2006

[4] F. Tecchia, C. Loscos, and Y. Chrysanthou. Visualizing crowds in real-time. Computer Graphics Forum, 21(4):753–765, December 2002a.

[5] S.Dobbyn, J.Hamill, K.O'Conor, and C.O'Sullivan, Geopostors: a realtime geometry / impostor crowd rendering system. In SI3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games, pages 95–102, New York, NY, USA, 2005. ACM Press.

[6] D.Gosselin, P.V. Sander, and J.L. Mitchell, Drawing a crowd. In Wolfgang Engel, editor, ShaderX3: Advanced Rendering Techniques in DirectX and OpenGL. Charles River Media, Cambridge, MA, 2004.

[7] N. Magnenat-Thalmann, H. Seo, F. Cordier, Automatic modeling of virtual humans and body clothing. In Proceedings of SIGGRAPH (2003), ACM Press, pp. 19–26.

[8] G. Ryder and A. M. Day, Survey of real-time rendering techniques for crowds. Comput. Graph. Forum, 24(2):203–215, 2005.

[9] S. Dobbyn, R. McDonnell, L. Kavan, S. Collins, and C. O'Sullivan, Clothing the masses: Real-time clothed crowds with variation. In Charles Hansen and D Fellner, editors, Proceedings of Eurographics Short Papers, pages 103–106. Eurographics Association, 2006.

[10] N. Magnenat-Thalmann, P. Volino, From early draping to haute couture models: 20 years of research. The Visual Computer, Springer, Vol. 21, No. 8, pp. 506–519, October 2005.

[11] R.McDonnell, M.Larkin, S.Dobbyn, S.Collins, and C.O'Sullivan, Clone attack! perception of crowd variety. ACM Trans. Graph., 27(3):1–8, 2008.

[12] B. Ulicny , P. De Heras Ciechomski, D. Thalmann, Crowdbrush: Interactie Authoring of Real-Time Crowd Scenes. In SCA'04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (New York, NY, USA, 2004), ACM Press, pp. 243-252.

[13] J. Pettré, P. de Heras Ciechomski, J. Maïm, B. Yersin, J.-P. Laumond, and D. Thalmann, Real-time navigating crowds: scalable simulation and rendering. Journal of Visualization and Computer Animation, 17(3-4):445–455, 2006.

[14] J. Pettré, H. Grillon, and D. Thalmann, Crowds of moving objects: Navigation planning and simulation. In Proceedings of IEEE International Conference on Robotics and Automation, pp. 3062-3067, 2007.

[15] J. Pettré, P. de Heras Ciechomski, J. Maïm, B. Yersin, J.-P. Laumond, and D. Thalmann, Real-time navigating crowds: scalable simulation and rendering. Computer Animation and Virtual Worlds, vol. 17, no. 34, 445–455, 2006.

[16] A. Treuille, S. Cooper, and Z. Popovic, Continuum crowds, Proc. SIGGRAPH 2006, pages 1160–1168, 2006.

[17] F. Morini, B. Yersin, J. Maïm, and D. Thalmann, Real-Time Scalable Motion Planning for Crowds, The Visual Computer, 24(10): 859-870, 2008.

[18] C.W. Reynolds: Steering Behaviors for Autonomous Characters, Proceedings of Game Developers Conference, San Jose, California, pp. 763-782, 1999.

[19] Soraia Raupp Musse and Daniel Thalmann. A hierarchical model for real time simulation of virtual human crowds. IEEE Transactions on Visualization and Computer Graphics, 7(2): 152–164, 2001.

[20] H. Grillon and D. Thalmann, Simulating Gaze Attention Behaviors for Crowds. Computer Animation and Virtual Worlds, Vol.3-4, 2009.

[21] A. Aubel, R. Boulic and D. Thalmann, Real-time display of virtual humans: levels of details and impostors, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 10, Nr. 2, pp. 207-17, 2000.

[22] I. Rudomin and E. Millan, Point based rendering and displaced subdivision for interactive animation of crowds of clothed characters. In VRIPHYS 2004: Virtual Reality

Interaction and Physical Simulation Workshop, pages 139–148, 2004.
[23] B.Yersin, J.Maïm, J.Pettré, D.Thalmann, Crowd Patches: Populating Large-Scale Virtual Environments for Real-Time Applications. Proceedings of I3D, 2009.
[24] J. Maïm, S. Haegler, B. Yersin, P. Mueller, D. Thalmann, L. Van Gool, Populating Ancient Pompeii with Crowds of Virtual Romans, Proc. VAST 2007, pp.109-116.