Proceeding of International Conference on Electrical Engineering, Computer Science and Informatics (EECSI 2014), Yogyakarta, Indonesia, 20-21 August 2014

Populating 3D Cities: A True Challenge

DANIEL THALMANN

Nanyang Technological University, Singapore & EPFL, Switzerland

ABSTRACT

In this paper, we describe how we can model crowds in real-time using dynamic meshes, static meshes and impostors. Techniques to introduce variety in crowds including colors, shapes, textures, individual animation, individualized path-planning, simple and complex accessories are explained. We also present a hybrid architecture to handle the path planning of thousands of pedestrians in real time, while ensuring dynamic collision avoidance. Several behavioral aspects are presented as gaze control, group behaviour, as well as the specific technique of crowd patches.

1. INTRODUCTION

Our main objective is to simulate the Virtual Life in cities of today or the Past. This means that we should be able to animate crowds of Virtual Humans inside a Virtual City. 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.
We can distinguish two broader areas of crowd simulations (S.R. Musse and D.Thalmann 2012). The first one is focusing on realism of behavioral aspects with usually simple 2D visualization, like evacuation simulators, sociological crowd models, or crowd dynamics models. In this area, a simulated behavior is usually from a very narrow, controlled range (for example, people just flying to exit or people forming ring crowd structures) with effort to quantitatively validate correspondence of results to real world observations of particular situations (P.A. Thompson and E.W. Marchant 1995). Ideally, simulation results would then be consistent with data sets collected from field observations or video footage of real crowds either by human observers (D. Schweingruber and C. McPhail 1999) or by some automated image processing method (A.N. Marana et al 1998). Visualization is used to help understanding simulation results, but it is not crucial. In most cases, a schematic representation, with crowd members represented by colored dots, or sticky figures, is enough, sometimes even preferable as it allows to highlight important information. In the second area, a main goal is high quality visualization (for example, in movies productions and computer games), but usually the development of an autonomous behavior model is not the priority. What is important is a convincing visual result, which is achieved partly by behavior models, partly by human intervention in the production process. A virtual crowd should both look well and be animated in a believable manner, the emphasis of the research being mostly on rendering and animation methods. Crowd members are visualized as fully animated three dimensional figures that are textured and lit to fit into the environment. Here, behavior models do not necessarily aim to match quantitatively the real world; their purpose is more in alleviating work of human animators, and to be able to respond to inputs in case of interactive applications.

Little work (S.R. Musse and D. Thalmann 2001; B. Ulicny and D. Thalmann 2002; W. Shao and D. Terzopoulos 2005; J. Pettre et al 2006) has yet tried to explore more general crowd models integrating several sub-components such as collision avoidance, path-planning, higher-level behaviors, interaction or rendering.

Realistic looking, believable behaving and real-time rendered virtual crowds are challenging. At the individual scale, virtual agent must look realistic, i.e. the 3D models are textured and lighted. They are goal directed behaving. People don't walk in the streets freely, i.e. going in a random direction each time they encounter an obstacle. Usually they walk with a goal in mind: going to work, shopping. At the crowd level, each virtual character should be unique. Except if you are surrounded by pairs of twins dressed similarly, in real life everybody has different morphology and clothes. Crowd dynamics should be respected. Virtual characters avoid each other to not collide. Flows are created naturally in a dense crowd. Another important aspect of crowd is the number of virtual characters. One can start speaking of crowd if at least one hundred agents are rendered. Massive crowds can count several thousands of characters. Real-time adds the constraint that virtual characters are simulated, animated and rendered at a frame-rate that allow user interactions.

For most crowd engines, individual characters are randomly walking around (J. Pettre et al 2006; S. Dobbyn et al. 2005). Much research was done on locomotion (F. Multon et al 1999; A. Treuille et al. 2007) and on object manipulation and grasping (N.S.Pollard and V.B. Zordan 2005). Such solutions work well for animating a single character but scale baldy for crowds. Even for a single virtual human, very little papers propose solutions to animate virtual humans that walk while carrying an object. K. Liu et al. (2005) use their dynamics and data based synthesis system to simulate an avatar walking with a weighted briefcase. A. Sulejmanpasic and J. Popovic (2005) adapt jump movement for a virtual human holding a briefcase. K. Yamane et al. (2004) propose a complex approach for virtual human to pick up boxes on a self and put them at another place but without locomotion. R. Heck et al. (2006) generate new animations by attaching upper-body actions of one motion to the lower-body locomotion of another motion. Thus, they are able to animate virtual humans carrying object while running or walking. R. Metoyer et al. (2007) synthesize virtual actors that can react to environment objects. Their actors keep balance after being hit by a ball. M. Sung et al. (2005) offer a solution to edit easily for a crowd what each individual has to do. In their demo, virtual agents were asked to carry boxes from one point to another.

2. GEOGRAPHICALLY-BASED INTELLIGENT BEHAVIOR

Crowd behavior is a paramount topic. Indeed, too often, crowd simulations are limited to virtual characters wandering in an environment, without any specific goal or interest. In order to remedy to this lack, we have worked on geographically-based behaviors for crowds. In other words, we change the behavior of crowds according to their position in space. This work has then been integrated in a close collaboration with the Computer Vision Laboratory of ETHZ to revive a district of Ancient Pompeii with crowds of virtual Romans (Figure 1) (J. Maim et al 2007).


Fig.1: Crowds of Virtual Romans populating Pompeii

In the crowd engine, motion planning is based on a navigation graph. Thus, at runtime, there is no direct interaction between the Romans and the city geometry. Instead, they interact with the navigation graph. In order to have them behave differently given their surroundings, the semantic data contained in the geometry need to be transferred into the navigation graph vertices.

We have designed an algorithm to read the low-resolution city model and automatically extract its semantic data to identify the influenced navigation graph vertices. In our specific scenario of Pompeii, we have developed two methods to identify the graph vertices which will adopt a special behavior. The first one, used for window and door semantics, returns all the graph vertices that are within a given distance to the concerned geometry (window/door quad). The second method, exploited for shop and bakery building footprints, detects all the graph vertices contained inside the footprint geometry. In our particular Pompeii case, the buildings all have simple footprints which allows to exploit a simple algorithm.

3. PATH PLANNING AND NAVIGATION

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 (J. Pettre et al 2006) a novel approach to automatically extract a topology from a scene geometry and handle path planning using a navigation graph. 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 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.


Fig.2: Crowd using hybrid path planning

For realistic motion planning for crowds, Treuille et al. (2006) proposed a method that 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

We proposed (F. Morini et al 2008) a hybrid architecture to handle the path planning of thousands of pedestrians in real time, while ensuring dynamic collision avoidance. Our improved short-term avoidance algorithm is based on the assumptions that pedestrians mostly want to first maximize their speed and second, to minimize detours. To efficiently avoid local inter-pedestrian collisions, this algorithm relies on a decomposition of the environment into a regular grid of square cells. It complements the potential field approach, which may fail when the available space is too small and too crowded.

To find pedestrians that can potentially collide, we take advantage of the grid structure covering the whole environment: at runtime, every pedestrian is registered in its current grid cell. In this way, we can reduce the search for possible collisions to a small set of neighbor cells. Although this simplification does not cut down the order of complexity in $O(n^2)$, it significantly decreases n , as compared to a brute force approach. 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. Figure 2 shows a crowd moving using the hybrid path planning algorithm.

4. GROUP BEHAVIOR

In our everyday life, it is rare to observe people in an urban scene walking all by themselves. Indeed, it is easy to notice that pedestrians often evolve in groups of 2 or more (F. Morini et al. 2007).. For this reason, we have introduced an additional and optional layer to our motion planning architecture. 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 (S.R. Musse and Thalmann 2001), 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 (J. Pettre et al. 2006), 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 2. Our approach is to use the Navigation Graph (Section 3) 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.

5. ACCESSORIES

Fig. 3: Simple 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.

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.

To allowed virtual humans to carry bag while the animation being still realistic, we proposed a new approach (B. Yersin et al. 2008). First we enhanced the accessories with additional information such as which joint of the skeleton need to be adapted and how. Joints could either be freeze or clamp. If freeze, at run-time the joint will be fixed to a certain angle value and override the animation for that joint. Such modification allowed virtual humans to hold an umbrella or a balloon. Joints could also be clamp. During animation, joint angles are limited to an interval. Such tuning, helps limiting the arm swing while walking with a shopping bag. It gives the impression that the object is heavy and limits oscillations. Such solutions are simple and efficient. However, when the camera is close enough artifacts appear. Some joint might be too stiff and the resulting motion could look robotic. Clamping joint angle limits the arm swing but in reality it should be scale down. With current implementation, at the extremity of the swing, the arm is stationary. Those artifacts break the level of realism of the crowd.

Fig.4: Crowds in groups with accessories

6. ANIMATION VARIETY

A second important factor, although less paramount is their animation. If they all perform the same animation, the results are not realistic enough (R. McDonnell et al 2008). 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.

Fig.5: PCA-based walking models

To generate our original set of walk and run cycles, we used the locomotion engine developed by Glardon et al. (2004); 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 5 shows an example.

7. 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. H. Grillon and D. Thalmann (2009) introduced 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.

Fig.6: An example depicting the types of possible gaze behaviors

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 6 shows virtual humans with gaze.

8. HOW TO IMPROVE THE PERFORMANCES

Several strategies and techniques for efficient **rendering of crowds** have been proposed as surveyed in (G. Ryder and A. M. Day 2005). Many solutions exploit the use of *impostors* to achieve high-performance rendering. Impostors (A. Aubel et al 2000) 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 (2004) have extensively used impostors for all instances, near or far ones, whereas (S. Dobbyn et. 2005) proposed to add static meshes in the vicinity of the camera and only using impostors at a farther distance.

Our approach (J. Maim et al. 2009) 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.

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 (B.Yersin et al. 2009). 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. CONCLUSIONS

We have identified the main problems to solve in order to simulate a virtual life in a 3D city. We have shown how we can make behaviors dependent on the Virtual Environment and how to develop strategies for path planning and navigation. We have also explained methods to create groups in crowds and to introduce simple and complex accessories and their impact on animation, especially locomotion and arm motionFinally, we have briefly discuss the problems of performance. The future of crowd simulation is surely in the interaction aspects. In the future, any person will be able to be immersed in a crowd and to interact with any member of the crowd. We have developed (Y. Wang et al 2013) such a prototype and it is promising.

10. REFERENCES

Aubel A., Boulic R., Thalmann D. (2000): 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.

Dobbyn S., Hamill J., O'Conor K., Sullivan C.O. (2005): Geopostors: a realtime geometry / impostor crowd rendering system, Proc. 2005 symposium on Interactive 3D graphics and games, ACM Press, New York, NY, pp.95–102

Glardon P., Boulic R., Thalmann D. (2004): PCA-based walking engine using motion capture data. In: Proc. of Computer Graphics International, pp.292-298.

Grillon H., Thalmann D. (2009): Simulating Gaze Attention Behaviors for Crowds. Computer Animation and Virtual Worlds, Vol.20. pp.111-119.

Heck R., Kovar L., Gleicher M. (2006): Splicing Upper-Body Actions with Locomotion, Computer Graphics Forum, Vol. 25, no. 3, pp.459-466.

Liu K., Hertzmann A., Popovic Z. (2005): Learning physics-based motion style with nonlinear inverse optimization, ACM Transactions on Graphics, Vol.24, No3, pp.1071–1081.

Maïm J., Haegler S., Yersin B., Müller P., Thalmann D., Van Gool L. (2007): Populating Ancient Pompeii with Crowds of Virtual Romans. The 8th symposium on Virtual Reality, Archeology, and Cultural Heritage, Proc. VAST '07, pp. 109-116.

Maïm J, Yersin B., Pettré J., Thalmann D. (2009): YaQ: An Architecture for Real-Time Navigation and Rendering of Varied Crowds, IEEE Computer Graphics and Applications, Vol.29, 4, pp.44-53

Marana, A. N., Velastin, S. A., Costa, L. F., Lotufo, R. A. (1998): Automatic estimation of crowd density using texture, Safety Science, vol. 28, no. 3, pp. 165-175.

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

Metoyer R., Zordan V.B., Hermens B., Wu C.C., Soriano M. (2007): Psychologically inspired anticipation and dynamic response for impacts to the head and upper body, IEEE Trans. Vis. Comput. Graph., 14(1):173-185.

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

Multon F., France L., Cani M.P., Debunne G. (1999): "Computer Animation of Human Walking: a Survey", Journal of Visualization and Computer Animation (JVCA), Vol. 10, pp. 39-54.

Musse S.R., Thalmann D. (2001): A hierarchical model for real time simulation of virtual human crowds, IEEE Transactions on Visualization and Computer Graphics, 7(2):152-164.

Musse S.R., Thalmann D. (2012): Crowd Simulation, Springer, 2nd edition.

Pettré J., de Heras Ciechomski P., Maïm J., Yersin B., Thalmann D. (2006): Real-time Navigating Crowds: Scalable Navigation and Rendering, Computer Animation and Virtual Worlds, Vol.16, No3-4, pp.445-456.

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

Rudomin I., Millan E. (2004): 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.

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

Schweingruber D., McPhail C. (1999): A Method for Systematically Observing and Recording Collective Action, Sociological Methods & Research, vol. 27, no. 4, pp. 451-498.

Shao W., Terzopoulos D. (2005): Autonomous Pedestrians, Proc. ACM/EUROGRAPHICS Symposium on Computer Animation, pp.19-28.

Sulejmanpasic A., Popovic J. (2005): Adaptation of performed ballistic motion, ACM Transactions on Graphics, vol. 24 (1), ACM Press, New York, pp. 165–179.

Sung M., Kovar L., Gleicher M. (2005): Fast and Accurate Goal-Directed Motion Synthesis for Crowds, Proc. ACM/EUROGRAPHICS Symposium on Computer Animation, pp.291-300.

Thompson P.A., E.W. Marchant E.W. (1995): A Computer-model for the Evacuation of Large Building Population, Fire Safety Journal , Vol. 24 , pp. 131-148.

Treuille A., Cooper S., Popovic Z. (2006): Continuum crowds, Proc. SIGGRAPH 2006, pages 1160–1168.

Treuille A., Lee Y., Popović Z. (2007): Near-optimal Character Animation with Continuous Control", ACM Transactions on Graphics 26(3), 7:1–7:7.

Ulicny B., Thalmann D. (2002), Towards Interactive Real-Time Crowd Behavior Simulation, Computer Graphics Forum, Vol.21, No4, pp.767-776

Wang Y., Dubey R., Magnenat-Thalmann N., Thalmann D. (2013): An immersive multi-agent system for interactive applications. The Visual Computer, 29(5): 323-332.

Yamane K., Kuffner J.J., Hodgins J.K. (2004): Synthesizing animations of human manipulation tasks, ACM Transactions on Graphics. 23(3):532–539.

Yersin B., Maïm J., Pettré, Thalmann D. (2009): Crowd Patches: Populating Large-Scale Virtual Environments for Real-Time Applications. Proceedings of I3D, pp. 207-214.

Yersin B., Maïm J:, Thalmann D. (2009): Unique Instances for Crowds. IEEE Computer Graphics & Applications, Vol.29, 6, pp.82-90.