Clothing the Masses: Real-Time Clothed Crowds With Variation

S. Dobbyn, R. McDonnell, L. Kavan, S. Collins and C. O'Sullivan

Interaction, Simulation and Graphics Lab, Trinity College Dublin, Ireland

Abstract

The animation and rendering of crowds of realistically clothed characters presents a difficult challenge in computer graphics, which is further exacerbated when real-time frame rates are required. To date, due to real-time constraints, standard skinning methods have been used to animate the clothes of individuals in real-time crowds, which does not create the appropriate secondary motion for flowing garments. However, plausible cloth simulation is vital for the depiction of realistic characters, so we have developed a novel crowd system in which the individuals are endowed with realism and variety through the addition of physically simulated clothing and hardware assisted pattern variation.

1. Introduction

Real-time crowd systems are increasing in popularity and importance, resulting in a greater need for more realistic virtual humans that populate them. Currently, such humans use skinned meshes for their clothing, often resulting in rigid and unnatural motion of the individuals in a simulated crowd. While there have been advances in the area of cloth simulation, both offline and in real-time, interactive cloth simulation for hundreds or thousands of clothed characters would not be possible with current methods. In this paper, we wish to address this problem and devise a system for animating large numbers of clothed characters.

The majority of games that implement cloth dynamics on current generation hardware do so in a highly constrained manner, often ignoring the issues of collision detection by allowing the cloth surface to penetrate nearby surfaces. Game developers favour cloth that is highly controllable and tend to use more traditional methods of bone based skinning and pre-simulated vertex mesh animations if the performance of the cloth is critical to the game. In simulated crowd scenes for games, cloth is rarely if ever used due to the large numbers of polygons required to accurately capture the deformation of the cloth. However, deformable clothing adds greatly to the realism of the characters.

We have added realism to our crowd simulations by dressing the individuals in realistically simulated clothing, using an offline commercial cloth simulator, and integrating this into our real-time hybrid geometry/impostor rendering system. Additionally, we present a technique for generating cyclical motion for pre-simulated cloth, which therefore moves in a fluid, realistic manner. Furthermore, we have added variety to our impostor representation by developing a new hardware rendering technique for adding pattern variety to the same cloth for different humans in a crowd. Our results show a system capable of rendering large realistic clothed crowds at interactive frame rates.

2. Background

Implementing realistic cloth dynamics in real-time game applications still represents a significant challenge for game developers. Simulating cloth deformation is a complex process both in terms of dynamics simulation and collision detection for the changing cloth shape. Many game developers have relied on more tractable but approximate solutions including the Verlet method [Jak01], and have restricted the simulation's complexity by only using a subset of the mesh to represent the cloth. Another method to introduce complex clothing is to generate the folds using cloth simulation and then use skinning to attach the clothing to the character. This method works well in some cases, but often results in the unrealistic bending of folds in the cloth, as the folds have to deform according to the skeleton of the character. This can make a long flowing skirt look like trousers with folds. Also, the important secondary motion of the cloth is lost in this case. In the animation research community, Cordier and Magnenat-Thalmann [CMT05] use a data-driven approach for real-time processing of clothes. Vassilev et al. [VSC01] developed an efficient technique for dynamic cloth simulation using a mass-spring model.

With real-time crowd systems, traditional rendering techniques become rapidly overwhelmed as a result of displaying many humans, and detail has to be sacrificed to achieve interactive frame rates. Previous work on rendering real-time crowds has focused on replacing complex virtual human models with simpler representations, such as lower resolution models containing fewer polygons [dHCSMT05] and impostors [TLC02] [DHOO05]. To avoid a crowd being populated with carbon copy clones, programmable graphics hardware has been utilised to add colour and animation variation to a virtual human model [DHOO05] [dHCSMT05] [GSM05].

3. Cyclical Cloth

In a real-time crowd system, the characters' animations are often cyclical in nature, so that they can be smoothly linked to allow them to move in a fluid manner. Cyclical animations are commonly obtained by manually altering the underlying skeletal motion so that they loop in a realistic looking manner. However, making looping animations using characters with pre-simulated clothing is a more difficult task, as manual cleanup of the cloth to make it cyclical is very time-consuming, particularly for very deformable items of clothing like skirts, and can result in unrealistic effects.

We wanted a more automatic way of generating cyclical cloth and began by creating a very long animated sequence, repeating the animation of the human many times and simulating the cloth in response to the repeating animation, in the expectation that it would at some point become periodic. On viewing these long sequences, it was found that the cloth did not always settle to a periodic state, particularly for highly deformable clothing such as long flowing skirts. A more robust method was needed in order to obtain a good cycle in the cloth. In a good cloth cycle, the cloth at the start frame Fs and at the end frame Fe of the animation cycle of length l should be the same, and be travelling at the same velocity. The long cloth sequence needed to be searched using a distance metric that took into account all of the vertices on the cloth mesh between the two frames of animation, in order to find one correctly cyclical loop.

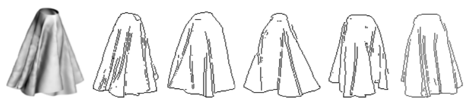

Figure 1: *Edge Images taken from 5 different viewpoints.*

We used a distance metric similar to Kovar et al. [KGP02] to compute the differences between all frames that were of length l apart in the sequence, and chose a set of candidate cycles whose distances were below a user set threshold. Usually, we chose the 5 cycles with the lowest distance metric as candidate cycles. In the case of stiff clothing, where the motion is very restricted, picking the cycle with the smallest distance metric was often enough to produce a good cyclical motion. For other more deformable, flowing clothing, this metric was often insufficient, as it did not weight the importance of the folds in the cloth, but rather, weighted all points equally.

Bhat et al. [BTH*03] showed that the human perceptual system is sensitive to moving edges, and used this to compare the folds and silhouettes of simulated cloth to that of video cloth sequences to find the difference between them. We based the second pass of our algorithm on this idea of matching folds and silhouettes, and devised a metric for comparing the candidate cloth cycles at Fs and at Fe. For each of the candidate cycles, we generated images of the cloth at Fs and Fe from 5 different viewpoints around the skirt. The images at Fs and at Fe for each of the viewpoints were then converted into edge images (Figure 1), using the standard Canny edge detection algorithm [Can86]. The mean distance between edges in the corresponding images of Fs and Fe were then found using an edge distance estimator, and the resulting differences in the 5 images were summed together, to give a final difference metric for the candidate cycle. The cloth cycle with the smallest edge difference was chosen as the final cycle. In most cases, this final cloth cycle was good enough for use, but in certain cases, an extra linear blend step between Fe and Fs was needed to produce the final cycle. We tried to avoid linear blending where possible, as it often resulted in the cloth intersecting with the human model, and we also felt that the results were more natural when blending was not used.

This method produced cloth that appeared cyclical from all viewpoints for all of the clothing that we tested. Cyclical appearance was judged by whether or not it was possible to notice a discontinuity in the cloth motion at the start and end of the cycle from all viewpoints (i.e., the looped animation did not exhibit a flicker in the cloth). We found that tighter clothing needed few animation cycles to produce cyclical cloth (sometimes as few as 4 cycles), as the cloth settled to a near periodic state quickly. Whereas looser cloth needed up to 40 animation cycles to produce a nice cyclical animation. The cyclical cloth and human animations could then be exported into a real-time system and replayed, and impostors were also generated as described in the next section.

4. Cloth Geopostors

Our crowd system is based on the approach of Dobbyn et al. [DHOO05], which uses a hybrid combination of image-based (i.e., impostor) and detailed geometric rendering techniques for skinned virtual humans. Furthermore, our system includes clothed characters by using commercial software [CloFX] to obtain our cloth simulations, but any high

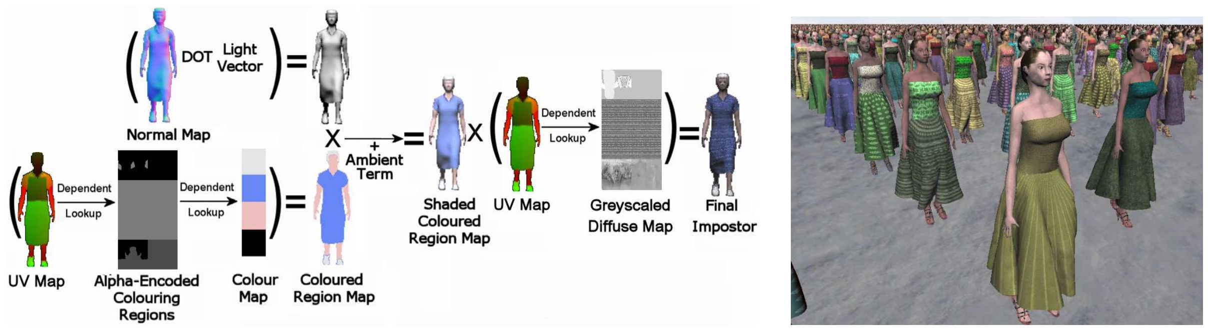

Figure 2: *(a) UV mapped impostor rendering sequence and (b) an example of adding variation to one character using 8 different diffuse textures.*

quality offline simulator could be used to produce the cloth animation. We pre-simulate the deformation of both the virtual human's skin mesh using linear blend skinning and its cloth mesh using the physical simulator, based on the motion of its underlying skeleton. However, while the secondary motion of the character's cloth greatly adds to our crowd's visual realism, cyclical cloth motion is necessary to avoid any jerky motion artefacts and we present a technique to solve this in Section 3.

Once the character's meshes are pre-simulated, they are then exported and stored in separate keyframed meshes or "poses". By pre-calculating and storing the deformation of the skin and cloth mesh in poses, this avoids the cost of deforming the character's body and simulating its clothes at run-time. Generating the impostor representation of our clothed character involves capturing two types of images from a number of viewpoints around the model: a detail map image to capture the detail of the model's diffuse texture and a normal map image whereby the model's surface normals are encoded as an RGB value. For more information on how these images are generated and rendered in our system see [DHOO05]. At run-time, the clothed virtual humans switch between the two level of detail (LOD) representations depending on their position with respect to the viewer.

Adding colour variation to an impostor representation has already been achieved through the encoding of colouring regions in the alpha channel of the detail image [TLC02] [DHOO05]. This is used at run-time to address a colour or "outfit" map through programmable texture addressing. In the case of a mesh, another method to add texture variation is to provide it with a set of different diffuse textures with which it can be texture mapped. However, applying this type of variation to the impostor would require exporting a set of detail maps for each different diffuse texture used, resulting in the rapid consumption of texture memory. To solve this problem, we propose replacing the detail map with a UV map.

5. UV Mapping Technique

We improve upon existing impostor techniques for adding variety by replacing the detail map images (described in Section 4) with a texture coordinate map or *UV map*. This is similar to a normal map whereby it is generated for each viewpoint and contains the texture coordinates of the model's surface encoded as a RGB value. At run-time, these values are used to lookup the same set of diffuse textures used by the mesh, allowing texture variation for both the human's skin and cloth. To also allow for colour variation, the alpha channel of the mesh's diffuse textures is encoded with alpha encoded regions which are used to lookup the colour map. The overall sequence for shading and adding both colour and texture variation to the impostor representation is shown in Figure 2. Before the impostor's UV mapped images are pre-generated, texture seams should be kept to a minimum when texture mapping the associated mesh. Otherwise, these seams result in incorrect texture coordinates being stored in the UV map, which causes rendering artefacts to arise at run-time due to the wrong pixels in the diffuse texture being addressed. A similar type of artefact also occurs when linear filtering is used, causing the background pixels and the impostor's silhouette to be linearly interpolated and generating incorrect texture coordinates. However, this is only noticeable when the impostor is close to the viewer.

This new type of image allows the texture variety and interest of each clothed impostor to be greatly increased. Replacing the detail map with the UV map image ameliorates the problem of trying to add the same type of variation using detail maps, which results in the consumption of large amounts of texture memory (see Section 6). Additionally, these UV maps could be further utilised to enhance the impostor's realism by applying various per-pixel lighting textures as specular maps, which are commonly used by high resolution game characters.

6. Results

Frame rate tests were carried out on the clothed crowd system to investigate how many clothed humans could be displayed using different LOD representations at 30 fps (see Figure 3). All of our tests were performed using a PentiumIV 3.6Ghz processor, with 2.0GB RAM and an ATI Radeon X850 XT Platinum Edition graphics card with 256MB of video memory. In each test, all of the humans were on the screen walking on the spot, and were dynamically lit.

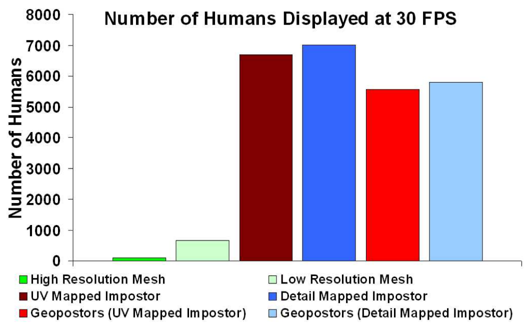

Figure 3: *Number of humans displayed at 30 fps using different LOD representations.*

It was found that 90 high resolution geometric models of clothed characters (13,056 triangles each) could be displayed onscreen at one time. With loss of visual quality, this number could be increased to 655 when low resolution geometry clothed characters were displayed (1,899 triangles each). Approximately 6,600 UV mapped impostors could be displayed at the same frame rate, but the closer humans appeared very pixellated. As expected, the number of detail mapped impostors displayed is higher due to the new UV map approach requiring extra texture lookups and per-pixel operations. When we used a hybrid geometry/impostor approach as in [DHOO05], up to 6,000 humans could be displayed (15 high resolution, 5,985 UV mapped impostors) which allowed visual quality and real-time performance to be maintained. Visual fidelity was maintained as impostors were displayed at the 1:1 pixel-to-texel ratio, where they are perceptually equivalent to high resolution meshes [HMDO05].

Furthermore, in the case of switching between a clothed character's mesh and impostor representation, adding variation to the cloth using the detail mapped approach requires a substantially larger amount of texture memory in comparison to the UV mapped impostor (see Figure 4). These calculations use a single clothed character and assume that the detail mapped and UV mapped impostor consist of 10 frames of animation, each normal map being a 1024x1024 sized RBG image and both the detail map and UV map are 1024x1024 sized RGBA images. Additionally, each diffuse texture used by the clothed character's mesh representation is a 1024x1024 RGBA sized image. DXT3 texture compression is used in these calculations to reduce the memory requirements by 4 for all RGBA images and by 6 for all RGB images.

7. Conclusion

Our technique of adding pre-simulated clothing improved the realism of our crowd system. The visual quality of our clothed crowd is maintained by creating cyclical cloth motions to avoid discontinuous motion artefacts. Additionally, the introduction of the impostor's UV map complements the use of impostors with its mesh representation in a LOD crowd system, since it allows the matching of texture and colour variation between the cloth and skin. This technique could be extended to add other realistic character effects such as pre-simulated hair or fur.

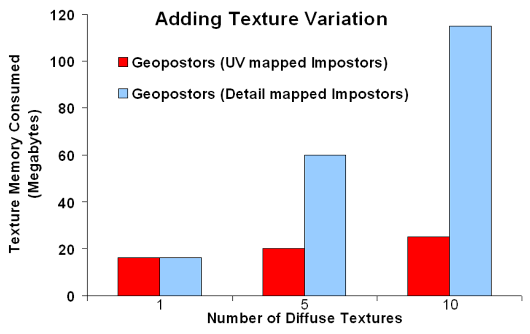

Figure 4: *Texture memory consumed by adding pattern variation to the Geopostor system using UV mapped and detail mapped impostors for a single clothed character.*

References

[BTH*03] Bhat K. S., Twigg C. D., Hodgins J. K., Khosla P. K., Popovic Z., Seitz S. M.: Estimating cloth simulation parameters from video. *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on computer animation* (2003), 37–51.

[Can86] Canny J.: A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI 8* (1986), 679–698.

[CloFX] Cloth simulation software, ClothFX.

[CMT05] Cordier F., Magnenat-Thalmann N.: A data-driven approach for real-time clothes simulation. *Computer Graphics Forum 24*, 2 (2005), 173–183.

[dHCSMT05] de Heras Ciechomski P., Schertenleib S., Maïm J., Thalmann D.: Reviving the roman odeon of aphrodisias: Dynamic animation and variety control of crowds in virtual heritage. *VSMM* (2005).

[DHOO05] Dobbyn S., Hamill J., O'Conor K., O'Sullivan C.: Geopostors: a real-time geometry / impostor crowd rendering system. *SI3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games* (2005), 95–102.

[GSM05] Gosselin D., Sander P., Mitchell J.: Drawing a crowd. *ShaderX3* (2005), 505–517.

[HMDO05] Hamill J., McDonnell R., Dobbyn S., O'Sullivan C.: Perceptual evaluation of impostor representations for virtual humans and buildings. *Computer Graphics Forum 24*, 3 (2005), 623–633.

[Jak01] Jakobsen T.: Advanced character physics. In *Proceedings of the Game Developers Conference* (2001).

[KGP02] Kovar L., Gleicher M., Pighin F.: Motion graphs. *ACM Transactions on Graphics 21*, 3 (2002), 473–482.

[TLC02] Tecchia F., Loscos C., Chrysanthou Y.: Visualizing crowds in real-time. *Computer Graphics Forum 21*, 4 (2002), 753–765.

[VSC01] Vassilev T., Spanlang B., Chrysanthou Y.: Fast cloth animation on walking avatars. *Computer Graphics Forum 20*, 3 (2001), 260–267.