Illuminating Crowds Via Integral Approximation of Cosine Lobes

R. Galvao
Email: R.Galvao@uea.ac.uk
School of Computing Science
University of East Anglia
Norwich, NR4 7TJ, UK

R. G. Laycock
Email: Robert.Laycock@uea.ac.uk
School of Computing Science
University of East Anglia
Norwich, NR4 7TJ, UK

A. M. Day
Email: Andy.Day@uea.ac.uk
School of Computing Science
University of East Anglia
Norwich, NR4 7TJ, UK

***Abstract*—Real time crowd rendering has long been the subject of intense research, which has led to many advancements in the field. The research up until now has either focused on increasing the number of virtual people rendered or on the reduction of the inherit homogeneity of real time crowds. With the rapid advances of technology, people's expectations of rendering realism have increased significantly. This means that simply rendering large varied crowds is no longer sufficient. Therefore more advanced rendering methods must be adopted so as to increase the realism of the rendered virtual people. This paper introduces a new illumination technique developed for the use in real-time crowds. This is achieved by using the approximations of cosine lobe integrals together with environment maps, allowing a more realistic rendering of crowds. The approximations are shown to have a low error, whilst maintaining interactive frame rates when used within a crowd. Furthermore, the rendering of the crowd is improved due to more realistic rendering approach which allows diverse BRDFs to be applied to the differing materials within each virtual human.**

I. Introduction

The dramatic advances in hardware over the last decade have fuelled increased realism on real-time crowd rendering. During the majority of this time most techniques have focused on increasing the number of rendered virtual humans and the increase in variety within the crowd. However, the illumination of such crowds still tends to be performed via the use of simple point based lighting systems, which often leads to unrealistic lighting of the crowds.

Outside of crowd rendering, many techniques have been developed to increase the illumination realism of renderings. They range from mimicking self shadowing via the use of Ambient Occlusion, to the use of irradiance maps, including many more. All of this derives from the fact that point lights are no longer sufficient. Further advances are needed for scenes to appear visually acceptable. Therefore, the current shift from point lights to more illumination algorithms is bound to increase with time.

Furthermore, the rendering of realistic crowds requires the usage of multiple materials. Each virtual human, potentially having its lighting influenced by static and dynamic elements of the virtual environment. Therefore, to realistically render a real time crowd, it is essential that the technique used be able to cope with multiple materials whilst its illumination is influenced by both static and dynamic objects.

II. Background

Over the last decades the use of virtual environments has increased exponentially. From games to archaeology, they allow a user to visit the impossible. Virtual environments need to be populated in order to look realistic. However, rendering large populations within a virtual environment, without a detrimental effect on the interactivity of the application can be challenging. The computational processing capabilities of computers are continuously increasing, with it our expectations for realism also increase. Over the last decade advances have been undertaken towards increasing the population sizes able to be rendered, which has now reached levels beyond required [5]. Since this point the research onto real time crowd rendering has shifted towards increasing variety within the crowd [2], [4]. However, the illumination models used are still based on simple point lights. Therefore it is essential to improve the illumination of real time crowds, so as to satisfy the increasing levels of expectation for real time renderings.

Every material reflects light differently. For any incoming light ray a differing amount will be reflected in each direction. This relationship is described by Bidirectional Reflectance Distribution Functions (BRDFs). Over the years a large number of BRDF models have been created [6], [7], [3]. Each of these has differing levels of realism and use of computational resources.

When used in real-time applications these approximations tend to only deal with light incoming from specific light sources. This, however, is significantly detrimental to the realism of the scene, since in reality, each object is illuminated by light incoming from all directions.

Whilst rendering realistic virtual crowds several requirements have to be taken into account. One of the most important is to allow a large variety of materials. Therefore, any BRDF representation used must have a low memory footprint, in order to allow for a larger quantity of materials to be stored. Since the [3] reflectance model, has an extremely low memory footprint per material, it proved to be a natural choice. The most significant underlying problem of this technique stems from the unintuitive nature of its coefficients, which poses a problem when artistic control of the lighting is required. However it is of little importance when used to replicate

measured BRDF properties.

The Lafortune reflectance model is described by Equation 1. It uses a summation of lobes to represent the reflectance properties of the material. Each lobe has three coordinate coefficients and a lobe power to represent it, so that only four terms are required to be stored per lobe. In Equation 1, C defines a three dimensional vector representing the three coordinate coefficients for a Lafortune Lobe, and w represents the power to which the cosine lobe is raised. The number of lobes used is described by the term L. The $\vec{u}$ and $\vec{v}$ vectors represent the incident and exitance direction vectors of the lighting calculation.

$$f(\vec{u},\vec{v}) = \sum_{i=1}^{L} \left[C_{x,i}u_x v_x + C_{y,i}u_y v_y + C_{z,i}u_z v_z\right]^{w_i} \quad (1)$$

This model allows the accurate calculation of the amount of light transmitted from an incident direction towards the viewer. However, for accurate reconstruction of an object's reflectance appearance all of the incident directions must be taken into account. Therefore this equation must be integrated over all possible incident directions.

The following sections, will propose techniques which will allow the approximation of this integral to be performed within a real-time application.

Within this work, the used Lafortune BRDFs were fitted to [1] BRDF data multiplied by the cosine of incident angle.

III. Illumination from Static Objects

To facilitate the integral approximation, the Lafortune model (Equation 1) can be rewritten as Equation 2.

$$\begin{aligned} &\vec{Cv}_i = \{C_{x,i}v_x, C_{y,i}v_y, C_{z,i}v_z\} \\ &f(\vec{u},\vec{v}) = \textstyle\sum_{i=1}^{L} \left(\hat{u}\cdot\hat{Cv}_i\right)^{w_i} \|\vec{Cv}_i\|^{w_i} \end{aligned} \quad (2)$$

The illumination influence of each lobe is constrained around the direction $\vec{Cv}_i$, with a radial size determined by w. Therefore, the average of the incoming illumination from the $\vec{Cv}_i$ direction with the appropriate radial distance is used. This approximation for the incoming light significantly reduces the needed lighting calculation. The problem is in obtaining the average light incoming from a certain direction and radial distance.

Using environment maps together with their MipMaps, allows easy access to averages of groups of texels. The important aspect of using this technique is to use an environment map representation which minimizes distortion. This is because the averaging of texels within a MipMap is performed in texture space. Therefore, lighting artefacts can be introduced by using an inappropriate environment map such as a spherical environment map. For this research cube environment maps were used, since they have native GPU support and exhibit a small amount of distortion.

This way it is possible to obtain the average light around $\vec{Cv}_i$ by obtaining it from an appropriate MipMap level, which is controlled by the value of w. By using this average of the incoming light per lobe, Equation 3 can be used as an approximation of the illumination. This equation introduces two functions: $m(\vec{a},b)$ gets the environment map value along direction a, on the MipMap level b (the direction being converted from tangent space to global space); $z(c)$ maps the cosine power onto the appropriate MipMap level, the resulting value being inversely proportional to the input. The term k_u being the amount of light incoming from direction u.

$$\int f(\vec{u},\vec{v})k_u du \approx \sum_{i=1}^{L} \|\vec{Cv}_i\|^{w_i} m(\vec{Cv}_i, z(w_i)) \frac{2\pi}{1+w_i} \quad (3)$$

Equation 3 provides a new way to approximate the integral of the illumination of an object by using the Lafortune BRDF representation. This provides an approximate illumination for light which is incoming from static objects.

However, since constantly updating an environment map would be too computationally expensive for most real-time applications, a different approximation is required to deal with dynamic objects.

IV. Illumination from Dynamic Objects

In order to obtain the illumination contribution from dynamic objects, a more approximate approach was taken. It is possible to approximate the illumination incoming from dynamic objects with a cosine lobe. This lobe has a direction vector in the direction of the dynamic object ($\vec{D}$). The power of this cosine lobe (t) will determine the radial influence of the incoming light. Therefore, its value will be determined by the size of the object divided by its distance.

In order to be able to approximate the integral of the multiplication of two cosine lobes, the first step is to determine the direction vector that obtains the peek result ($\vec{MV}$).

$$\hat{MV} = \hat{D}t^{0.7} + \hat{Cv_i}w^{0.7} \quad (4)$$

Equation 4 was developed via empirical analysis to approximate the direction vector for the peek result. This provides a highly accurate direction vector with low error, when the angle between $\vec{Cv}_i$ and $\vec{D}$ is acute. However, if the angle is obtuse Equation 5 must be used instead.

$$\begin{aligned} &\hat{CP} = \hat{Cv_i} \times \hat{D} \\ &\hat{Vc_1} = \hat{CP} \times \hat{Cv_i} \\ &\hat{Vc_2} = \hat{D} \times \hat{CP} \\ &\hat{MV} = \hat{Vc_1}t^{0.7} + \hat{Vc_2}w^{0.7} \end{aligned} \quad (5)$$

Once the direction of the peek vector is obtained, it can be plugged into Equation 6 in order to obtain its length.

$$\|\vec{MV}\| = h\left(\hat{MV}\cdot\hat{Cv_i}\right)^{w} h\left(\hat{MV}\cdot\hat{D}\right)^{t} \quad (6)$$

With this, it is then possible to obtain a rudimentary approximation of the integral (Equation 7).

$$\int f(\vec{u},\vec{v})du \approx \sum_{i=1}^{L} \|\vec{Cv}_i\|^{w_i} \frac{\pi\|\vec{MV}\|\left(\hat{D}\cdot\hat{Cv_i}+1\right)}{1+w+t} \quad (7)$$

Since the static illumination is used in combination with this technique, it is necessary to first remove the light contribution that is blocked by the dynamic object. To do this, simply obtain the amount of light from the environment map in the direction of the $\vec{MV}$ vector. This value can be multiplied by the result of each lobe for Equation 7. This value has then to be subtracted from the illumination for the surface.

To obtain the illumination being reflected by the dynamic object, it is necessary to estimate the amount of light that this dynamic object is obtaining. By assuming the object is facing the opposite way to the surface being illuminated. Simply negate the x and y components of the $\vec{MV}$ vector, and use it to sample the environment map. This value can then be multiplied by the result of each lobe for Equation 7. The result of which is added to the surface illumination

V. Applying it to Virtual humans

The described dynamic illumination technique is based upon spherical objects. In order to apply this to more complex geometry, the shape of such objects has to be approximated by proxy spheres.

To accurately approximate an avatar using proxy-spheres, a large number of spheres would be required. This would significantly detriment the performance of the proposed illumination algorithm, due to the large quantity of illumination dynamic illumination elements. Therefore in a pre-processing stage the most influencing bones in terms of self illumination are determined for each point on the surface of the avatar.

A. Avatar pre-processing

The first stage of avatar pre-processing the bones are analysed against their influencing vertices to determine the size of spherical blocker needed. The size of proxy sphere used for each bone is obtained, by analysing the distances to the vertices it is associated with. Then each vertex of the avatar is checked against the avatar's skeleton for every keyframe of animation. The bones of the skeleton that most block the illumination to the vertex are saved, and their Ids are placed within the vertex data that is passed to the GPU.

B. Avatar Rendering

During rendering the currently most influencing bones must be obtained for each vertex, which is performed in the vertex shader. Each vertex is compared against the few bones which were determined most influencing in the pre-processing stage (Figure 1). Within this implementation the calculation of the skeletal animations are dynamically calculated within the GPU. Therefore the necessary positional data of the avatar's skeleton is already present within the GPU, and easily accessible by the vertex shader. The position of each vertex is projected, using a dot product, to the bones obtained in the pre-processing stage. The projected point is then used as the centre of the proxy sphere. The generated proxy spheres are then compared and the most influential in terms of illumination is passed onto the fragment shader. Within the fragment shader the received proxy-sphere is then used as a dynamic illumination object as described in Section IV.

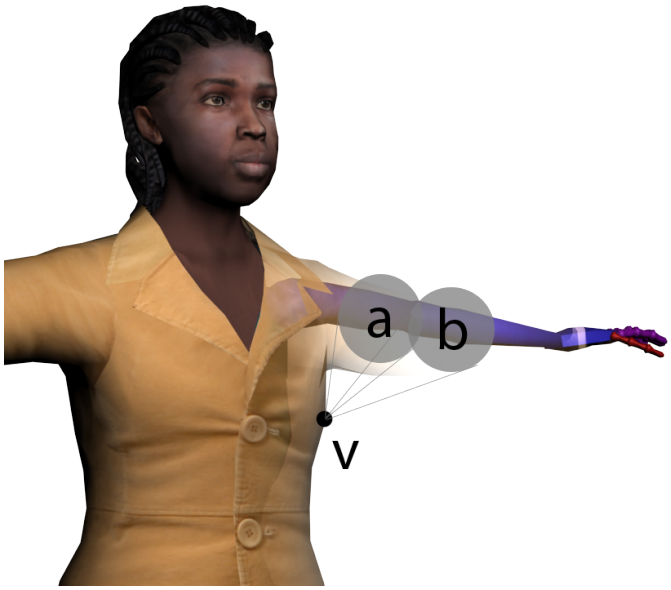

Fig. 1. The bones which most influence vertex *V* are represented by the spherical blockers *a* and *b*.

VI. Crowd Rendering

Within a crowd of people the illumination of each avatar is not only influenced by itself, it is also influenced by the avatars around it. In order to efficiently approximate this, levels of detail for the proxy-sphere approximations of the avatars must be employed. In the lowest level of detail the entire avatar can be approximated by a single proxy-sphere. In the highest level of detail, the approach taken is similar to the self occlusion described in the previous section. Since the avatars are similar to each other in structure the most influencing bones of the other skeleton can easily be predicted and compared in the vertex shader. After the most influencing proxy-sphere in the skeleton of the other avatar is determined it is passed along onto the fragment shader. To further increase performance, only the closest avatars are used to influence each other's illumination.

In order to also allow for BRDF variety for the differing materials within the crowd, the material Ids already in use for generation of colour variety obtained, and differing Cosine Lobe Coefficients are assigned for each material ID of each avatar. To select coefficients for each avatar whilst maintaining realism, appropriate materials are randomly selected from predefined lists of materials.

VII. Populating Streets

The most common use of crowds is to populate urban streets. However, the described illumination technique is dependent on environment maps. Therefore, a two level environment map system has been used. The first level of the environment maps contains the environment maps captured at street level; the second contains the sky map used for the scene. For the street level environment maps, all the static objects within the scene are captured with the exception of

the sky map. Where there would have been sky on these environment maps the alpha value of the texture is set to zero.

Several street level environment maps are captured along each street and saved onto a texture array. This allows each avatar to dynamically select the closest environment map based on its location within the street.

The discreet locations at which these environment maps are captures, means that natural variations of illumination, due to changes in position, are mostly lost. To solve this, the structured nature of city streets can be used in order to perform the appropriate parallax effect on the sampled environment map when moving away from the position from which the map was captured.

Streets tend to have buildings on both sides running in parallel, a floor below and sky above. Therefore the streets can be thought as being composed of three planes, if the sky is ignored. To facilitate the application of an environment map parallax, it is recommended that the street level environment maps be captured at a constant pre-defined height, and the capturing be performed aligned with the buildings and ground.

In order to perform the parallax, the environment map sampling vector is extended until it hits one of the three planes. Then the shift in position is added to the sampling vector. The resulting vector becomes the sampling vector used on the street level environment map. Due to this parallax, the sampled mipmap must also be shifted. Therefore if the vector increases in size the sampled mipmap must be lowered and if the vector size decreases the sampled mipmap level must be increased.

After sampling the street level environment map, if the alpha value is below one, the sky map is sampled using the original sampling vector. The returned value is then multiplied by the inverse alpha value of the street level environment map, and both environment maps sampled are then added together. The combination of these two sampled environment maps is then used to calculate the illumination as described in previews sections.

VIII. Results

The presented technique was tested on a Dell XPS, with a NVidia GeForce 8800 GTX graphics card, and Intel core 2 extreme CPU. Using a screen resolution of 1680x1050. The performance test scene was composed of 1000 virtual humans, each composed on average of 6500 polygons. The environment being composed of 100000 polygons, whilst illuminated by an HDR sky map, and using the variety techniques presented by [2].

From testing, it was possible to observe, that although the presented technique has a noticeable performance impact, it still allows a large crowd to be rendered at a high definition whilst maintaining interactive frame rates of over 24 frames per second.

Figure 2 demonstrates two avatars influencing each others illumination.

As a supplement to this paper, a multimedia attachment is provided in the form of a video. This video demonstrates the illumination of virtual humans being dynamically influenced by their own movements, and the movements of other virtual humans.

Fig. 2. Illumination of two virtual humans, influencing each others illumination.

IX. Conclusion and Future work

This paper has presented techniques to improve the illumination of real-time crowds, by approximating the integral of cosine lobes, together with environment maps. The presented illumination technique provides a new way to increase visual realism of real-time crowds whilst maintaining interactive frame rates.

For the future there are several lines of research available. Although the presented algorithm already has a high performance, its efficiency can be further improved by reducing texture accesses within the GPU. Artists tend to prefer to use other BRDF models due to the unintuitive nature of the Lafortune Model. Therefore, another line of research is to look at other BRDF representations, to find techniques to approximate their integration.

References

[1] CuRRET (1999) Curret: Reflectance and texture database (columbia university). http://www.cs.columbia.edu/CAVE/software/curet/

[2] Galvao R, Laycock R, Day A (2008) Gpu techniques for creating visually diverse crowds in real-time. In: VRST, Bordeaux, France, pp 79–86

[3] Lafortune EPF, Foo SC, Torrance KE, Greenberg DP (1997) Non-linear approximation of reflectance functions. In: SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp 117–126

[4] Maïm J, Yersin B, Thalmann D (2009) Unique character instances for crowds. IEEE CG&A 29(6):82–90

[5] Millan E, Rudomin I (2006) Impostors and pseudo-instancing for gpu crowd rendering. In: GRAPHITE '06, ACM, New York, NY, USA, pp 49–55

[6] Phong BT (1975) Illumination for computer generated pictures. ACM, New York, NY, USA, vol 18, pp 311–317

[7] Schrder P, Sweldens W (1995) Spherical wavelets: Efficiently representing functions on the sphere. In: Siggraph 1995