International Journal of Computer and Electrical Engineering, Vol. 2, No. 1, February, 2010
1793-8163

LOD based Real Time Shadow Generation for Subdivison Surfaces

Mustafa, S. Fawad, Member, IACSIT

***Abstract*—Shadow generation in gaming and animation industries has attracted many researchers to achieve the best possible outcome from their concepts for complex scenes. Realism to the Scene is only possible if we have accurate shadows with complex objects. For this reason Subdivision surfaces were introduced to describe a surface using a polygonal model with fine smoothness. Here, we discuss an approach to efficiently render shadows for subdivision surfaces by introducing the Level-of-Details and visibility function. In this way we can achieve fast and robust scene rendering with multiple subdivision models. In this paper we have proposed an improvement on rendering rate of shadow volume technique for subdivision surfaces using graphic hardware.**

Index Terms—shadow volume, geometric images, Level-of-Details, graphics hardware

I. Introduction

Geometry Images have become a powerful tool not only because it process geometry faster and uses less memory than "polygon soup" but also because it has also became an integral and inevitable part for various diversified research areas. This may include algorithms of mesh simplification, deformation, global illumination [17], shadow generation [15] and dynamic LOD technique [12].

The fast rendering of a virtually realistic scene has always been the main motivation behind many researches. GPUs have contributed a lot in fast rendering through their ability of processing parallel to the CPU and their computation power.

Subdivision Surfaces have seen loads of researches in the recent years mainly after its use by Pixar for animating *Geri* in 1998 [18]. They have been a favourite among the researchers and developers due to their ease of usage and implementation. Subdivision Surfaces have been successfully evaluated through GPU by different techniques. Shiue et al [2] presents a real-time surface evaluation by using fragment shaders, Boubekeur et al. [19], on the other hand, uses a vertex shader to generate mesh refinement on GPUs. Shadow generation in subdivision surfaces has seen very little research. Initially, it was modeled by converting the surface into facet models and then applying the techniques of shadow generation through any shadow rendering technique but Tang et al. [16] proposed a technique to generate volumetric shadows which was later revamped in Tang et al. [15] by introducing geometry image to it.

Subdivision Surface supports dynamic LOD by using lesser number of subdivisions. Dynamic LOD has been an additional attraction to the subdivision surface as it lessens the number of polygons for computation but when used in Tang et al.'s [15] technique it loses this property and becomes limited.

In this paper, we have proposed a view-dependent LOD which brings this support of dynamical LOD back to the algorithm making it more efficient. Since, the main allure of Tang et al. [15] is that it produce a purely GPU accelerated shadow volume algorithm so we have proposed an algorithm which adds itself in the previous algorithm without much fuss and retains it as a GPU accelerated algorithm. There are different approaches to implement dynamic LOD in geometry through GPU that may include [5] which uses a quadtree like structure to produce a seamless geometry image atlas and [12] which uses a mipmaps to produce dynamic LOD structure for geometry images. We use a method which can be considered as an extension to the method proposed in [12].

II. OUR APPROACH

The algorithm proposed is a fully graphics hardware accelerated method and encompasses all the steps of rendering from creating a geometry image of model to a rendering of the scene with shadows.

For the sake of simplicity, we have subdivided the algorithm into phases based on the different fragment shaders it uses and it can be illustrated as follows:

In precomputation, we generate the geometry image from the control mesh of the subdivision surface. Then it generates geometry image for different level of details including the geometry image of original resolution. Then for selection criteria, a selector map is devised. The process flow has been shown in Figure 1 below.

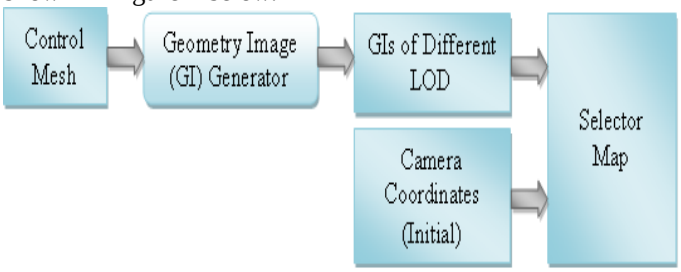


Figure 1 Precomputation

A. Generation of Geometry Images

The geometry image is evaluated by the analytical method described by [15] as:

$$S_{k,n}(u,v) = C_0^T \, Weight(u,v) \quad (1)$$

where,

$S_{k,n}(u,v)$ is the surface evaluated by limit position at a given resolution,

C_0^T is the initial control mesh &

$Weight(u,v)$ are the terms independent of the control mesh and specific to parameters (u, v).

Evaluating the input control mesh and weights through a fragment shader provides two geometry images one having limit positions and other having normal vectors.

B. Mipmap Generation

For LOD, the FBO is used to generate mipmap by scaling the model by half and translating it in a way to be at the centre of the origin and converting it into a geometry image [12], thus, removing the chance of undersampling. The minimum resolution of a mipmaped geometry image is reduced to 8 x 8 and maximum resolution is '(3/2 width) x height', where width and height is of original geometry image.

C. Selector Map Generation

Next step, evidently, is to define a criterion for the selection of geometry image of particular resolution, and for this a selector map is created and stored in memory for further use. The map is created by dividing the maximum distance from the camera to the object [12]. The map generated allocates a range of distance to each geometry image according to the level of detail provided in it.

This selector map is, then, used at run-time to select a geometry image for the given resolution.

The precomputation is done for all the models in the scene. Since, the whole process is computed within GPU and only the control mesh is passed on by CPU which greatly improves the speed. The geometry images of different resolution are stored in FBO and selector map is stored in a separate floating point FBO.

After this, at run time we will select one geometry image of suitable resolution depending on the distance of the object from camera. The flow of this phase is shown in Figure 2.

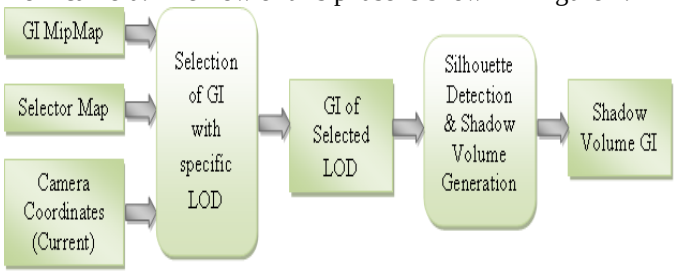


Figure 2 Runtime

D. Selection of Geometry Image of Specific LOD

The geometry images of different resolution, the selector map and the current camera coordinates are used at runtime for selecting a geometry image of particular LOD. The geometry image is selected by calculating first, the distance between the camera and object and then using the selector map to decide which geometry image should be used amongst images of different LODs.

E. Silhouette Detection & Shadow Volume Generation

The selected geometry image is then used further for detection of silhouette to produce a silhouette image. Since, the method used for silhouette detection in [15] uses the vertices first to check their visibility and our method does not disrupt the basic profile of the object from any angle so the silhouette would not have any cracks. The silhouette image produced by this process is then used for shadow volume generation. The shadow volume generation and silhouette detection can be done by using the same method proposed by [15] so we are not discussing it any further. As a result of this computation a new geometry image is formed which is actually the shadow of the geometry image of specific resolution. If the object is very far from the camera then a shadow will be formed of very small resolution. This way, not only the computation of the original image is greatly reduced but also the shadow required for this image will require less time to generate image. It can be noticed that if the original image is of minimum resolution i.e. '8 x 8' then the silhouette image will be of '16 x 8' and the image size of the shadow volume will be '32 x 8'. This is evident from this calculation that the speed will improve further if the object is away from the camera.

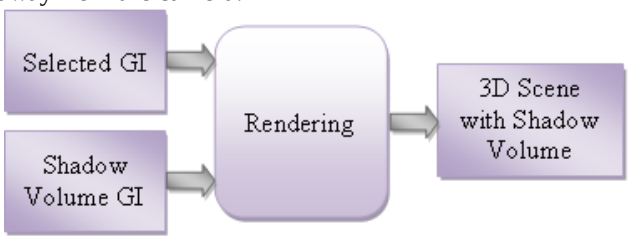


Figure 3 Rendering

The resultant images are then passed through another fragment shader for final rendering of the scene with volumetric shadow of the models as shown in the Figure 3.

III. Results

The results are generated on a Windows XP system with NVIDIA Geforce 7200 GS card. We have expanded [15]'s work to accommodate our algorithm by storing different LODs of the same model. For the sake of simplicity we have used only one model, here for demonstration, which has 288 patches.

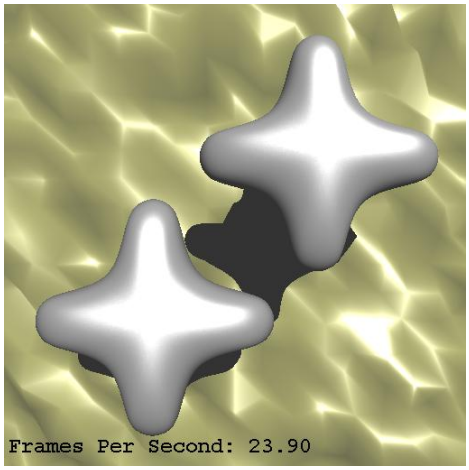


Figure 4 Rendering with full detailing

It can be easily noticed that the real-time rendering of [15]'s algorithm is seriously affected if more models are introduced to the scene as shown in the Figure 4. Here it is shown that with this addition of model the frame rate has been reduced to half compared with [15].

By reducing the level of detail in the scene we can easily restore the frame rate. This is shown in Figure 5 with half the level of detailing of the original model and in Figure 6 with minimum possible detailing.

It can be noticed that Figure 6's models has a visible change in the model but when we are far away from the model this change is not evident to human eye. Thus, by exploiting this fact we can easily render the scene with much more frame rate and with almost the same results.

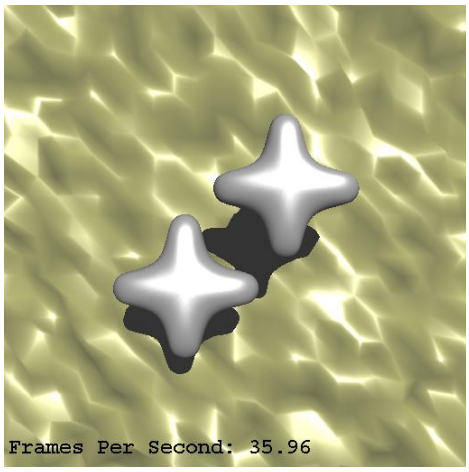


Figure 5 Rendering model with half amount of detailing

These three examples are the extreme cases of the rendered scene with maximum, medium and minimum level of detail respectively. This is done by reducing the size of Geometry Image as shown in Figure 7, 8 and 9 which shows original mesh's Geometry Image, the Geometry Image of normals and Geometry Image of the volumetric shadows, generated by [15]'s method, respectively.

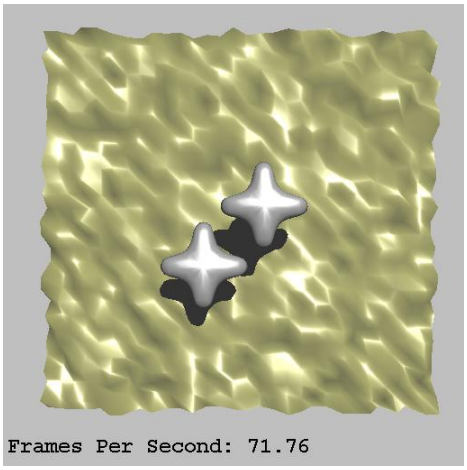


Figure 6 Rendering model with minimum amount of detailing

Level of detail	Texture		Rendering rate (fps)
	Width	Height	
Maximum	198	176	~24
Moderate	108	96	~35
Minimum	36	32	~72

Table 1 Results

Table 1 summarizes the results of the cases discussed above. Maximum level of detail corresponds to the conventional method provided by [15] and shown in Figure 4. The moderate method is the one when the user can see a little of the detailing but is far away to notice small changes as show in the Figure 5. The minimum LOD is used when the user is far away to notice detailing, it is also noticeable that the model with minimum level of detailing is de-shaped but since the user is far away so it cannot be noticed. Table 2 shows the comparison of different LODs with increasing number of models in the scene. Maximum is the conventional method using all the possible detailing provided by the original model.

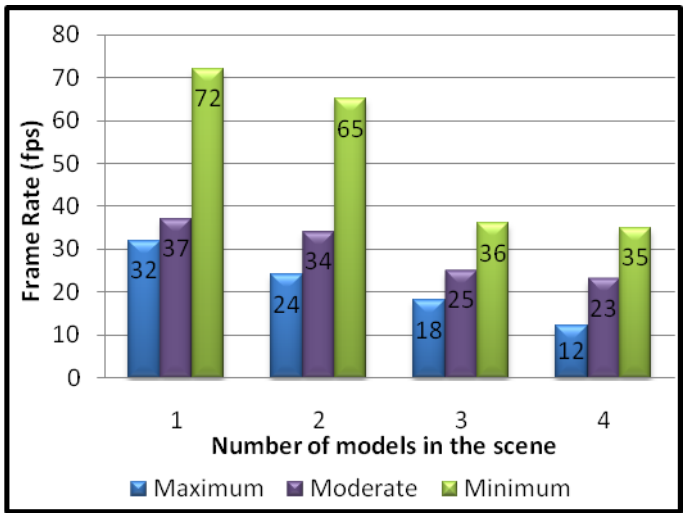


Table 2 Frame Rate with different number of models

Figure 7, 8, and 9 show the geometry images of the original mesh, its normals, and the geometry image of the

volumetric shadow created through the silhouette of the model as described by [15] used in the scenes in Figure 5, 6, and 7 respectively. The geometry image on the left of figures is for the maximum LOD of the model. The geometry image in middle is for moderate and evidently the geometry image on the right is for minimum LOD of the model.

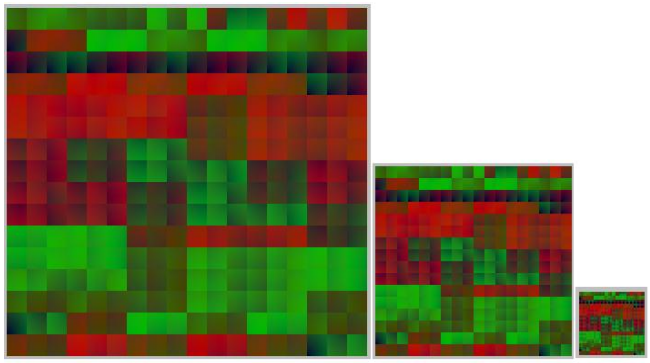


Figure 7 Geometry Images of mesh with different LOD

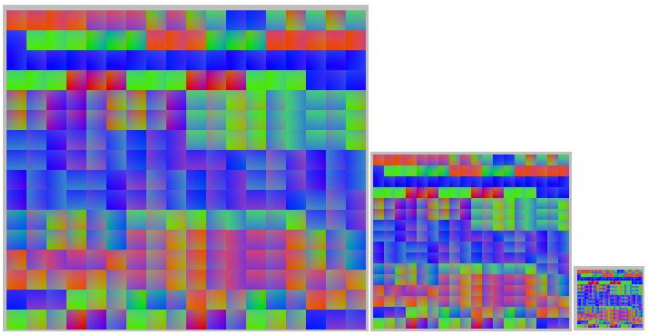


Figure 8 Geometry Images of normals with different LOD

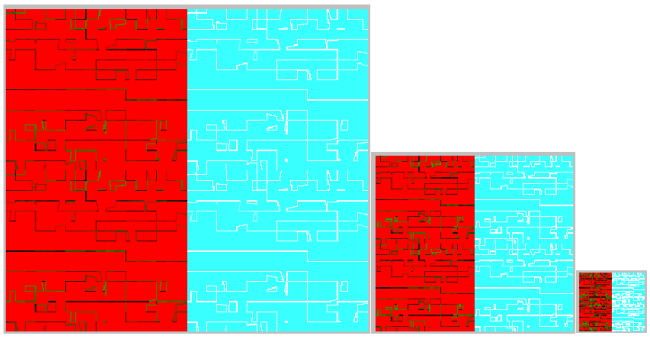
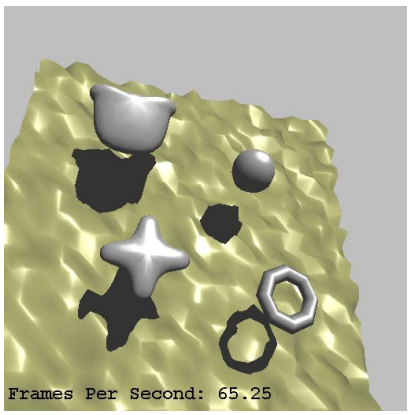
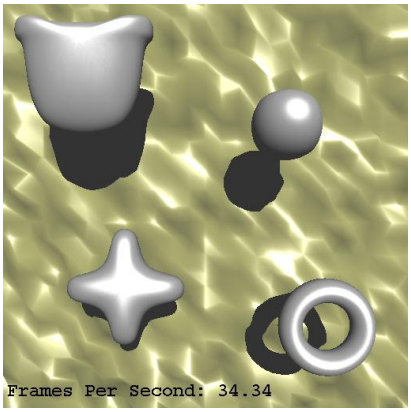


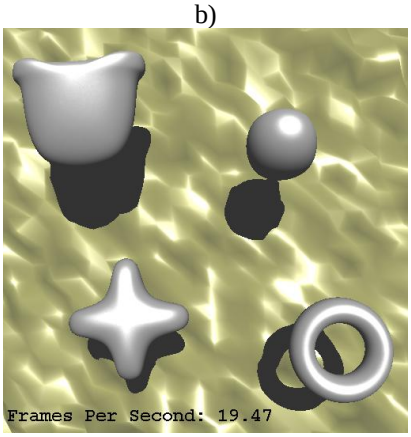
Figure 9 Geometry Image of shadow volume with different LOD



a)



b)



c)

Figure 10 Different LODs scene with 4 subdivision surface models, where a) has the minimum LOD, b) has moderate and c) has maximum possible detailing for all models.

Figure 10 shows a scene with 4 different models. This is to show that our scene can accommodate as many models as one wish. All are rendered through our method i.e. with different LODs for different camera distance where the uppermost is with minimum possible detailing for all models. All the models are modeled through subdivision surfaces with model on upper left corner has 288 patches, on upper right has 96 patches, lower left model has 288 patches, and lower right model has 32 patches. The frame rate of conventional method is ~20 fps, whereas our method gives ~35 fps for moderate detailing and ~65 fps for minimum possible detailing.

IV. Conclusion

Here, we have shown that fast shadow rendering for subdivision surfaces depending on viewer's position and Geometric Image (GI) size reduction can be achieved. Also, we can see the opportunity of working further with subdivision surfaces for gaming industries. For further results refer to Figure 10.

For future work, we are looking into possibility of introducing soft shadows and multiple area light sources. Dynamic scene with moving lights could also pave path for further research into subdivision surface properties.

V. Acknowledgement

Mustafa, S. Fawad would like to thank Dr. Zhaojun Liu, School of Information Science and Engineering, Shandong

University, China, for his valuable suggestions throughout the research work.

REFERENCE

[1] XIANFENG GU, STEVEN J. GORTLER and HUGUES HOPPE, "Geometry Images", In ACM Transactions on Graphics, Volume 21, 2002.
[2] SHIUE, L-J., JONES, I., and PETERS, J. 2005. "A Realtime GPU Subdivision Kernel." In Proceedings of SIGGRAPH Computer Graphics Proceedings, 1010– 1015.
[3] HOPPE, H. 1996. "Progressive meshes." In SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, ACM Press, New York, NY, USA, 99–108.
[4] HOPPE, H. 1997. "View-dependent refinement of progressive meshes." In SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 189–198.
[5] JI, J., WU, E., LI, S., and LIU., X. 2005. "Dynamic LOD on GPU". In Computer Graphics International, 108– 114.
[6] LOSASSO, F., HOPPE, H., SCHAEFER, S., and WARREN, J. D. 2003. "Smooth geometry images". In Symposium on Geometry Processing, 138–145.
[7] LUEBKE, D., REDDY, M., COHEN, J.AND VARSHNEY, A. W. B., and HUEBNER, R. 2002. "Level of Detail for 3D Graphics." In Computer Graphics and Geometric Modeling. Morgan Kaufmann Publishers, San Francisco, CA.
[8] BUNNELL, M. 2005. "Adaptive tessellation of subdivision surfaces with displacement mapping". In GPU Gems 2, Addison Wesley, 109–122.
[9] SANDER, P. V., WOOD, Z. J., GORTLER, S. J., SNYDER, J., and HOPPE, H. 2003. "Multi-chart geometry images". In Symposium on Geometry Processing, 146–155.
[10] TARINI, M., HORMANN, K., CIGNONI, P., and MONTANI, C. 2004. "PolyCube-Maps". In ACM Transactions on Graphics 23, 3 (Aug.), 853–860. Proceedings of ACM SIGGRAPH 2004.
[11] XIA, J. C., and VARSHNEY, A. 1996. "Dynamic view-dependent simplification for polygonal models." In VIS '96: Proceedings of the 7th conference on Visualization '96, IEEE Computer Society Press, Los Alamitos, CA, USA, 327–ff.
[12] HERNANDEZ, B., and RUDOMIN, I. 2006. "Simple dynamic LOD for geometry images." In GRAPHITE '06: Proceedings of the 4th international conference on Computer graphics & interactive techniques in Australasia & Southeast Asia.
[13] STAM, J. 1998. "Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values." In Proceedings of the 25th Annual Conference on Computer Graphics and interactive Techniques SIGGRAPH '98. ACM, New York, NY, 395-404.
[14] E. CATMULL and J. CLARK, "Recursively Generated B-Spline Surfaces on Arbitrary Topological Meshes", In Computer Aided Geometric Design, Vol. 10, Issue 6, 1978.
[15] MIN TANG and JIN-XIANG DONG, "Geometry Image-based Shadow Volume Algorithm for Subdivision Surfaces", In Proceedings of the Computer Graphics International, 2007.
[16] MIN TANG, JIN-XIANG DONG, and SHANG-CHING CHOU, "Real-Time Shadow Volume Algorithm for Subdivision Surface Based Models", In Proceedings of the Computer Graphics International, 2006.
[17] CARR, N.A., HOBEROCK, J. CRANE, K., and HART J.C. 2006. "Fast GPU ray tracing of dynamic meshes using geometry images." In Proceedings of Graphics Interface. A.K. Peters, 2006.
[18] DEROSE, T., KASS, M., and TRUONG, T. 1998. "Subdivision surfaces in character animation". In Proceedings of the 25th Annual Conference on Computer Graphics and interactive Techniques SIGGRAPH '98. ACM, New York, NY, 85-94.
[19] BOUBEKEUR, T. and SCHLICK, C. 2005. "Generic Mesh Refinement on GPU", In Proceedings of ACM SIGGRAPH/ Eurographics Graphics Hardware.
[20] LUEBKE, D. P., and ERIKSON, C. 1997. "View-dependent simplification of arbitrary polygonal environments". In SIGGRAPH, 199–208.

Syed Fawad Mustafa, is BEngr (Computer Systems) from Hamdard University, Pakistand and MSc (Computer Science) from Univeristy of Saarland, Germany.