LOD Exploitation and Fast Silhouette Detection for Shadow Volumes

Mustafa S. Fawad, Wang Wencheng, Wu Enhua

***Abstract*—**Shadows add great amount of realism to a scene and many algorithms exists to generate shadows. Recently, Shadow volumes (SVs) have made great achievements to place a valuable position in the gaming industries. Looking at this, we concentrate on simple but valuable initial partial steps for further optimization in SV generation, i.e.; model simplification and silhouette edge detection and tracking. Shadow volumes (SVs) usually takes time in generating boundary silhouettes of the object and if the object is complex then the generation of edges become much harder and slower in process. The challenge gets stiffer when real time shadow generation and rendering is demanded. We investigated a way to use the real time silhouette edge detection method, which takes the advantage of spatial and temporal coherence, and exploit the level-of-details (LOD) technique for reducing silhouette edges of the model to use the simplified version of the model for shadow generation speeding up the running time. These steps highly reduce the execution time of shadow volume generations in real-time and are easily flexible to any of the recently proposed SV techniques. Our main focus is to exploit the LOD and silhouette edge detection technique, adopting them to further enhance the shadow volume generations for real time rendering.

***Keywords*—**LOD, perception, Shadow Volumes, Silhouette Edge, Spatial and Temporal coherence.

I. INTRODUCTION

SHADOW generations have always been a major factor in 3d rendering. They play the important role in making the 3d scene more realistic. They define the distance of the object in the scene with respect to light. Shadows are also useful for a variety of other reasons: First, they help understand relative object placement in a 3D scene by providing visual cues. Second, they dramatically improve image realism and allow the creation of complex lighting ambiances. Recently many ideas are flourished considering how to simplify the calculations of shadow polygons in shadow volumes. Shadow Edges after their projection to the receiver have been studied more in detail. Considering the importance of shadows in 3d graphics, shadow volumes have come along way making its valuable place ingenerating shadows. To generate shadow volumes (SVs) detection silhouette edges at the boundary of the occluders are very important. Simpler the edges at boundary, easy are the generation of shadow polygons. SVs have always been of serious research consideration when the heavy-densed meshes or complex models are used. Above all, the main issue to generate fast robust shadows starts from the very first step. To support this task we would like to have complex but rather simple objects. The way to have this approach, we have explored different level-of-details techniques that can be studied to simplify the heavily dense meshes using to our requirement without disturbing the shape of silhouettes much to project shadows.

On the other hand, silhouettes play an important role in shape/boundary recognition because they provide the main cues for the figure to ground distinction. The brute-force approach to finding the silhouette edges simply checks every edge of the mesh, every frame. This may suffice for high-quality, non-interactive animations, which can afford to sacrifice speed for guaranteed results, but causes a major bottleneck in real-time applications.

In our paper, we have discussed the way to detect and track [15] the silhouette edges comparatively faster than brute force method and showed that even the simplified version of the complex objects could help is in precise shadows by exploiting the human perception. Our method has two part; 1) we generate the simplified version of heavily dense meshed objects as accepted by the human eyes for the lesser number of silhouettes. 2) then we apply the Markosian et al [5]'s silhouette edge detection method edge detection and silhouette tracking method for the next frame once the position of the light moves, taking advantage of spatial and temporal coherence to further boost up the rendering time for shadow generation [15].

Later, we apply the simple shadow volume method to show that this way we can achieve high speed edge detection and tracking for newer silhouettes for new frame in much faster way than the orthodox methods.

Mustafa S. Fawad is with the Key Laboratory of Computer Science, Institute of Software, Graduate University of Chinese Academy of Sciences, Beijing – 100039, China (phone: +86-10-88255778; email: fawad@ios.ac.cn)

Wang Wencheng is with the Key Laboratory of Computer Science, Institute of Software, Graduate University of Chinese Academy of Sciences, Beijing –100039, China

Wu Enhua is the Head of Computer and Information Science Department, University of Macao, Macao, China

Key Lab of Computer Science, Chinese Academy of Sciences, China and University of Macau, Macau, China supported this Project, under the Ph.D. fellowship awarded to first author of this paper

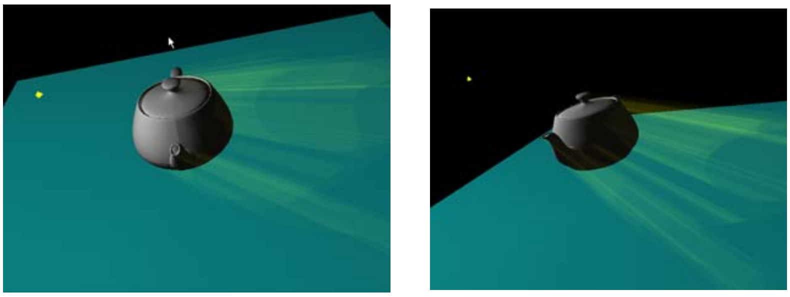


Fig. 1 Left: Original mesh; Right: Simplified mesh of Teapots

In our paper, we have referred to the relevant works and promote our idea on those studies. We, then, have explained our method and showed some results. Finally, we have concluded by suggesting some future works and advantages of exploiting these ideas to different existing shadow volume.

II. RELEVANT WORK

Many researchers have studied human perception and shadow generation together. Most of the works have been experimental. Hu et al [2] and Madison et al. showed the importance of shadow as visual clue for the object-to-object contact.

Kersten et al [4] investigated that the motion of the object and shadows generated for spatial perception. Similar approach with little diversity was experimented by Wanger et al [10]. He investigated the context of object size, spatial position, shadow shape and sharpness of simple objects. Shadow Volume generation is the classical approach made by Crow et al [1]. Later many robust solutions were proposed. The recent SV approaches were promising towards effective shadow generations.

Where we talk about shadows, silhouette edges play the most crucial and highly dependent role deciding the boundaries of the shadow. Gooch et al [13] and Benichou et al [12] presented a preprocessing procedure based on projecting face normals onto a Gaussian sphere. Here, every mesh edge corresponds to an arc on the Gaussian sphere, which connects the normal's projections of its two adjacent polygons. Applying this observation to silhouette edge extraction removes the need to check for each frame if every face is front or back facing.

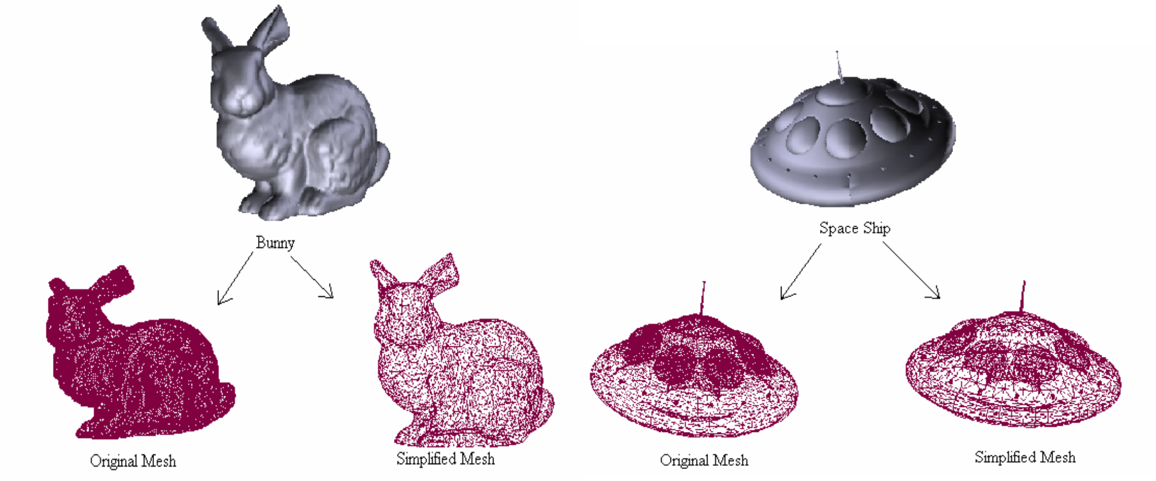


Fig. 2 Original and simplified versions of models using LOD technique

These techniques adopted the pre-computed nature which limits its adoption to shadow volumes because run-time expense could not be reduced by precomputation, since shadow volumes like shadow maps are determined with respect to a specific arrangement of lights and occluders.

Brute force algorithm is always a simple approach in this scenario. But once the fast detection of silhouettes and accuracy comes in, some intellectual handling of silhouettes is required. Work in this area has been done in vast range for non-photorealistic image rendering or computer vision. Later it was found out that they are highly productive to photorealistic approach. This idea inspired us to experiment one of the promising approach by Markosian et al [5 and 6].

III. OUR APPROACH

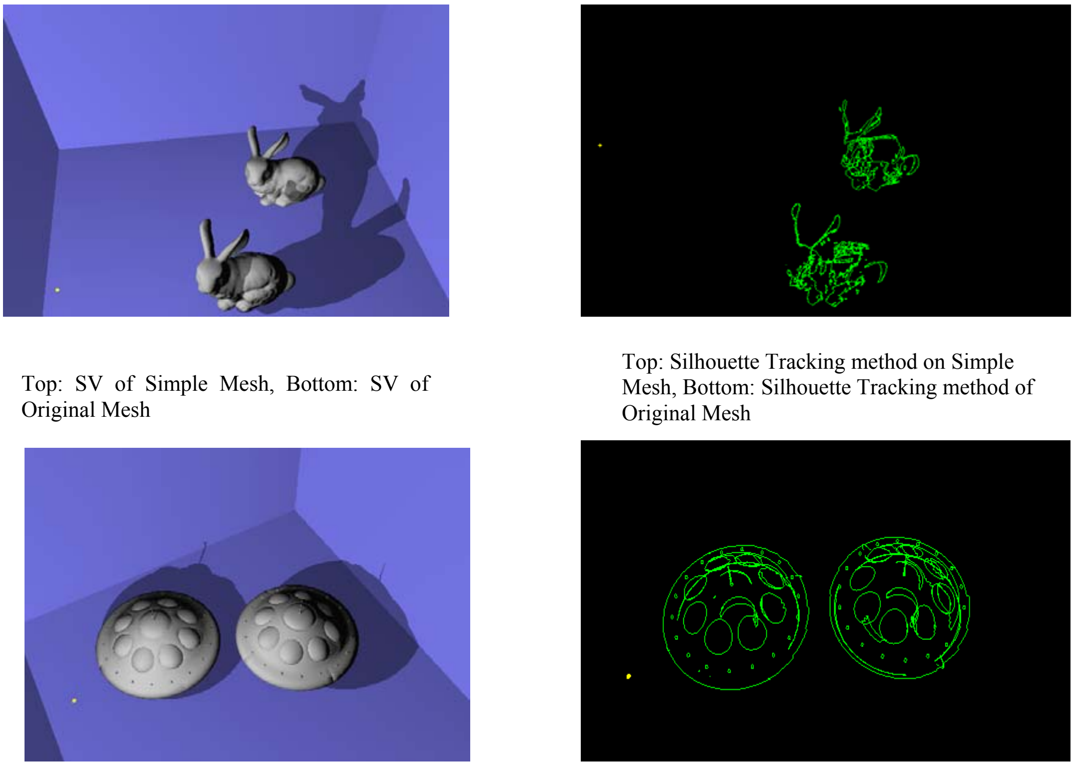
To keep the simplicity we use general shadow volume algorithm to extrude the edges to have bounding volumes of shadows. Meanwhile, the initial phase of our algorithm uses the following steps to make the silhouette edge computation so affordable as far as cost is concerned.

We take the advantage of perceptual exploitation of human eye to simplify the polygonal objects to improve the runtime performance of silhouette edge detection of generally medium and complex objects by generating their level of details LOD (See Fig2.).

Then from various silhouette edge detection methods including Canny's method, we use the highly effective Markosian et al.'s method silhouette edge detection method. In contrast to some pre-computation methods, this stochastic method can also gain faster run time execution.

A. *Simplification of the Polygonal model*

The simplification method has been of great advantage to fast real time rendering. LOD adoption has proved to be very useful when the time comes to show something similar but less complex for models. Human eye has always been less sensitive to shadows as compared to their own models, precisely, when shadows are made for only showing the good effects. Here, we use the LOD method to check which level is suitable enough to produce good results for hard shadows using point light source. Various LOD techniques can be used to test our technique.



Top: SV of Simple Mesh, Bottom: SV of Original Mesh

Top: Silhouette Tracking method on Simple Mesh, Bottom: Silhouette Tracking method of Original Mesh

Left: SV of Simple Mesh, Right: SV of Original Mesh

Left: Silhouette Tracking method on Simple Mesh, Right: Silhouette Tracking method of Original Mesh

Fig. 3 Bunny and Space Ship models using Markosian's [5 & 6] Silhouette tracking Method [15]

To remain with basics, we use ***VIZup*** software for effective LOD generation and loaded handful of simplified versions of models. Adopting the concept of Sattler et al [8], we choose of different versions of the model. The reason to adopt this idea is that the shadow of even the quite complex object is always simple and is completely concern with boundary edges. Keeping in mind this, we use simplified version, which negligibly changes the boundary edges and exploiting this behavior we have chosen the most simplified version based on their approximately exact shadow of original mesh. Later, with this simplified version we will use the silhouette tracking method mentioned below.

Models for the experiments are Bunny, teapot and Spaceship. We use simple shadow volume method to see which LOD of the meshed object is good enough to produce exact shadow (Human perception exploitation). As for distance, it is twice between the object and the plane, since this allows a wide rang of viewing angles from object and shadows are fully visible.

During the experiment, the test person is able to move the light source and the point of view around the object, while the viewing distance is fixed. Thus, it is possible to examine the generated shadows under several viewing angles. The current LOD can also be changed interactively. We have also test different levels with the silhouette-tracking algorithm explained in *B* and found promising results.

B. Silhouette Edge Extraction:

Here we introduce the implementation [15] of Markosian et al [5 and 6] ‘s method to detect the edges for shadow volume generation deliberately trading accuracy and detail for speed. By using this method we observed that the only few edges in the polygonal model are the actually silhouette edges. To find the initial set of candidate edges for front and back face culling, we randomly select the small fraction of edges and the exploit the idea of spatial and temporal coherence to only compute the new edges for the next frame, speeding up the rendering of the scene.

To start the algorithm we require the most important edge connectivity table. The table maps each polygon in the model to its directly adjacent polygons. The plane equation of polygons is also kept.

The algorithm is also promises to perform well only two polygons sharing each edge (closed mesh). We also highlight the way to deal with open meshes cases [15]. The table is simple when we deal with the closed mesh, but once the mesh is not close then we keep the information of border edges as well.

To track the behavior of silhouette between two different frames, we need to know the possible cases that may occur. They could be the different distances between the light and the object, which includes the following situations:

When silhouette is larger than the previous frame (small distance).

When silhouette is smaller than the previous frame (large distance).

When silhouette changes shape with change in position (similar distance).

In all Fig. 3 cases, its easy to find the silhouette edge [15] from previous once by picking an edge based on 1) either of the two adjacent faces if visible with one face visible to the light position. 2) if both are visible and one is visible to the light, silhouette edge further away from light is to be checked and 3) if both are invisible also to the light, silhouette edge nearest to light is to be checked.

Several algorithms can be followed to choose the path from one edge to another in order to find the silhouette edge. We use the left to right approach to walk on the objects models faces to find the silhouette edge. Since the second vertex of the last edge would always be same to the first edge of the first vertex of the silhouette (a kind of circle), it’s easy to track a silhouette and its behavior. Infinite rounds on circles can be avoided by capping the iterations. Following is the simplest yet efficient code of such tracking Tom et al. [15]:

```
FUNC find_edge_by_walk
            (Face FromFace, unsigned int EdgeIndex)
LET CurrentFace be NextFace
LET int BackEdge be NextFace->GetEdgetoFace(LastFace)
           LET StepLeft be true
           WHILE (not found)
          //check adjacent edges to see if they are silhouettes
LET LeftSideFace be NeighbourFace to CurrentFace on
      edge (BackEdge+1)%3
LET RightSideFace be NeighbourFace to CurrentFace on
      edge (BackEdge+2)%3
  IF LeftSideFace is non existent or !visible to the light
                  RETURN silhouette edge
//edge shared between LeftSideFace and CurrentFace
  ELSE IF RightSideFace is non existent or ! visible to
            light
    RETURN silhouette edge
 // edge shared between RightSideFace and CurrentFace
ELSE
IF StepLeft
  LET CurrentFace be LeftSideFace
 // i.e BackEdge be edge of LeftSideFace adjacent to
   CurrentFace
  ELSE
   LET CurrentFace be RightSideFace
//i.e BackEdge be edge of RightSideFace adjacent to
  CurrentFace

        END IF
LET StepLeft be NOT StepLeft
END IF
END WHILE
```

C. *Silhouette Edge Tracking:*

Since all silhouettes are closed loops, once a single edge to a silhouette is found, all edges to the silhouette (with respect to light position P) can be found by using the following algorithm Tom et al. [15]:

```
            Algorithm Complete Silhouette
            FUNC(Face VisibleFace, unsigned int EdgeIndex)
            Append Edge(VisibleFace, EdgeIndex) to the
            silhouette list
            LET CurrentFace be VisibleFace
             LET BackEdge be EdgeIndex
              WHILE TRUE
               LET TestEdge be (BackEdge+2)%3
                LET TestFace be adjacent to CurrentFace on
               edge TestFace
                 IF TestFace is non existent or is not visible to
                     the light
              IF TestFace equals VisibleFace and TestEdge
                    equals EdgeIndex
                           //test to see if we have completed the
                            //silhouette loop
                            BREAK
                 ELSE
                   //we have found a silhouette edge
Append Edge(TestFace ,TestEdge) to the silhouette list
                 LET BackEdge be TestEdge
                 END IF
                ELSE
              //iterate to the next face
LET BackEdge be edge of TestFace adjacent to
        CurrentFace
            LET CurrentFace be TestFace
            END IF
            END WHILE
```

During completion, this algorithm is effective as it adopts the visible faces bordering the silhouettes. As the resulted silhouettes are ordered, the rendering gets optimized. Generally we take in account two cases: 1) concave silhouettes, made up of two or more edges that are concave on the model and 2) convex silhouettes, made up of edges other than concave behavior. The advantage of concave silhouettes is that they can be easily found and concave edges can be easily tracked [15].

To further optimize these silhouettes, we can easily look for new silhouettes on concave angles that are only greater than the certain degree (>=190 degrees). Simultaneously if we keep the ordered list of such models from greater angle to least, this can optimize the runtime [15].

In this case if ideal situation is made (i.e. no boundary edges) then the algorithm is highly and solely inspired by Markosian's [5 and 6] randomized silhouette detection method. Taking advantage of temporal coherence, this method uses the silhouettes found in the previous frame as a starting point for a search of the current frame. For a mesh with n edges, we randomly select a small fraction of edges to test [15]. When a new silhouette is found, its neighbors are also checked for local continuation of the silhouette contour, leveraging the spatial coherence of silhouettes. In our experience, this algorithm has proven efficient and robust enough for our real-time applications.

Though, apparently the method is more supportive to concave silhouettes, we can easily adapt to convex silhouette (if the mesh is not closed) detection by maintaining the list of border edges in the beginning and then add the check for new silhouettes in the border list if the any of the face is visible. Fig.3 shows that the shadow volume generation of original and simplified model seems similar to human eye, which indeed is a great advantage of human perception exploitation.

IV. RESULTS

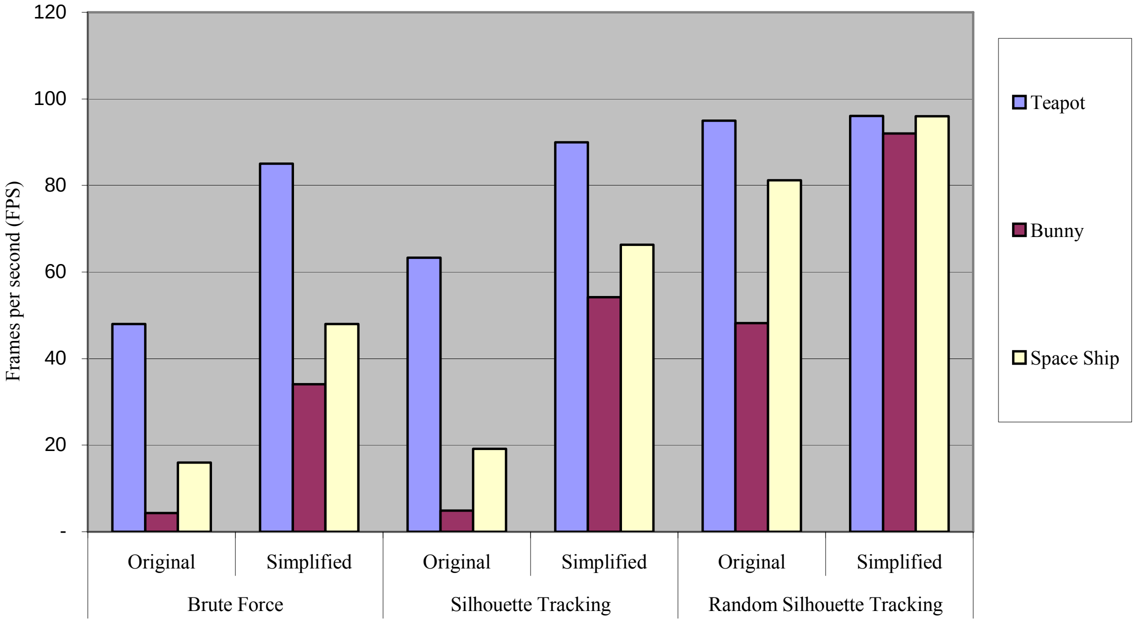
Different observers were tested for shadow perception; as a result a very much-simplified polygonal mesh was selected for complex objects resulting in faster silhouette detection. (see Table I). Here, we observe that if silhouette edge detection is tuned to highest concave angle, No edge type is checked and no boundary edges, the algorithm will act as the Markosian's randomized silhouette detection method [5 6 and 7] and Tom et al.[15]speeding up the tracking (see Table II and Fig.4). Results are as follows:

TABLE I
NUMBER OF VERTICES AND TRIANAGLES
FOR SOME TESTED MODELS

Tested Models	Original Mesh		Simplified Mesh*	
	Vertices	Triangles	Vertices	Triangles
Teapot	3,644	6,320	1,657	3,160
Bunny	34,834	69,451	3,532	6,945
Space Ship	10,016	19,608	3,179	6,078

TABLE II
COMPARATIVE RESULTS WITH OUR LOD AND SILHOUETTE DETECTION METHOD

Tested Models	Silhouette Detection Method					
	Brute Force (FPS)		Silhouette Tracking (FPS)		Random Silhouette Tracking (FPS)	
Meshes	Original	Simplified	Original	Simplified	Original	Simplified
Teapot	48.00	85.00	63.30	90.00	95.00	96.10
Bunny	4.35	34.09	4.90	54.20	48.20	92.00
Space Ship	16.00	48.00	19.10	66.30	81.20	96.00



Interestingly it is observed that, simpler objects like teapot do not require much more simplification as it will start showing inaccurate boundaries but nevertheless some simplification still possible having approximately same shadows. On the other hand, it is also the case of complex models which are highly dependent on their detailed meshes that further simplification to 50% to 70% cannot be done to have the general outer shape of the object (Including models which share more than two triangles for edges e.g. trees). Where as, closed models like bunny and space ships give more promising results long their boundaries for the projection of shadows with highly simplified version. For all the objects, Markosian's randomized method can be used where speed is more important then exact accuracy of the shadow boundaries.

We observe that, with Randomized method, output is still acceptable and results in high speedup factor as shown in the Table I and II.

V. CONCLUSION AND FUTURE WORK

Finally, we conclude that taking advantage of human perception on shadows and with LOD implementation we can generate the shadow volumes for comparatively less dense and simplified polygonal mesh to that of heavy meshes. We have shown that the observer is too flexible on shadow boundaries rather than actual mesh boundaries. We have also taken in account the hardcore professionals as observers and built but our conclusion that still we can have immense amount of reduction in triangles of heavily dense models. Further more, we have exploited one of the silhouettes tracking technique proposed by Markosian et al [5 and 6] and have taken the advantage of spatial and temporal coherence [15], overall increasing the running time and rendering quality of shadows on the boundaries of heavily meshed polygons.

Our approach is another step to approximation, which may lead to some artifact special when the edge is shared by more than two triangles or not a closed model.

Our work is more precisely experimental and noticeable with the fact that even the techniques of LOD generations and silhouette detection techniques for non-photo realistic images combined using point light source for hard shadows, can be very much advantageous to shadow volume. Keeping in mind the fact that silhouette detection and extrusion to shadow volumes plays important part in shadow volumes, as a future work, we still would like to explore and test other silhouette extracting and tracking techniques for non- photo realistic rendering and LOD methods. For LOD methods, one way is to have exact simplified matches using Hausdorff error computations. For Silhouette edge detections, we would like to consider some smooth edge detection techniques for curve surfaces or seek the advantage of spatial coherences with various other methods proposed. These steps could really improve execution time and time space trade offs. Here, we have shown the results based on hard shadows, which can be extended to soft shadows giving us more room to simplify the polygonal mesh of the object further exploiting the blurring effect of boundaries of the generated shadows. Our future aim is to investigate other approaches which can be easily singled-out by well illustrated tutorial of Luebke et al[14].

Although, we had some good results to improve the initial steps to further enhance simple shadow volume algorithm but it also shows that it can be really used to already other Robust SV algorithms. Our aim is to introduce the new way to further focus on combination of reality of human eye perception and see how promising the different silhouette edges techniques, summarized by Isenberg et al [3], can be for boundary edge detection before generating shadow polygons for shadow volumes. We promote this study because once the soft shadows step in using area light sources, their blurriness property will allow us to simplify the object further. Even, for the hardliner observers it would be acceptable.

Various other approaches have paved way through silhouette edge techniques and can be considered to further enhance the robust rendering of shadow volumes. Our further aim is to adopt this idea to graphic hardware. Stefan Brabec et al [9]. has some promising results with GPU while working with silhouette edge techniques. Besides his contribution, some other non photo-realistic silhouette edge techniques are a part of future research where we would explore the flexibility of these methods to photo-realistic rendering. Some hybrid approaches where image precision techniques can be studied is now fairly wide open for research. In this aspect, Raskar et al [11] has provided some results. Isenberg et al [3]‘s survey on silhouette edge detection techniques is a better guide for us to adopt the suitable approach on the basis of our requirement. Our aim would be explore brighter ways from it.

In the end we conclude that different LOD algorithms and Silhouette algorithm have proved to be most promising for model simplification and edge detection with efficiency and accuracy. Hence, their exploitation could be an open research in finding the robust, faster and far better solution.

ACKNOWLEDGMENTS

We would like to thank Ulf Assarsson, Assistant for sharing his innovative and valuable suggestions.

REFERENCES

[1] Crow, F. C., “Shadow algorithms for computer graphics”. In SIGGRAPH: Proceedings of the 4th annual conference on Computer graphics and interactive techniques, ACM Press, 242–248, 1977.

[2] Hu, H. H., Gooch, A. A., Thompson, W. B., Smits, B. E., Shirley, P., and Rieser, J. J., “Visual cues for imminent object contact in realistic virtual environments”. In IEEE Visualization 2000.

[3] Isenberg T, Freudenberg B., Halper N., Schlechtweg S., Strothotte T., “A Developer's Guide to Silhouette Algorithms for Polygonal Models.” Journal of IEEE Computer Graphics and Animation,. 28-37, 2003.

[4] Kersten, D., Knill, D., Mamassian, P., and Bulthoff, I. “Illusory motion from shadows.” Nature 379, 31, 1996.

[5] Markosian, L., Kowalski, M. A., Goldstein, D., Trychin, S. J., Hughes, J. F., and Bourdev, L. D. “Real-time nonphotorealistic rendering”. In Proceedings of the 24th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co., 415–420, 1997.

[6] Markosian, L., “Art-based Modeling and Rendering for Computer Graphics.” PhD thesis, Brown University, May 2000.

[7] Markosian, L., Barbara J. Meier, Michael A. Kowalski, Loring S Holden, J. D. Northrup, and John F. Hughes. “Art-based Rendering with Continuous Levels of Detail.” In Proceedings of the First International Symposium on Non Photo realistic Animation and Rendering (NPAR) for Art and Entertainment, June 2000.

[8] Sattler M., Sarlette R., Mucken T. and Klein R., “Exploitation of human shadow perception for fast shadow rendering.” In Proceedings of the 2nd symposium on Applied perception in graphics and visualization, 2005

[9] Barbec, S. and Seidel H.P. “Shadow Volumes on Programmable Graphics Hardware.”, In Computer Graphics Forum, Volume 22, 2003.

[10] Wanger, L. “The effect of shadow quality on the perception of spatial relationships in computer generated imagery.” Proceedings of the 1992 symposium on Interactive 3D graphics, ACM Press, New York, NY, USA, 39–42, 1992.

[11] Raskar, R. and Cohen, M. “Image Precision Silhouette Edges.” In Symposium on Interactive 3D Graphics (I3DG), Atlanta, April, 1999

[12] Benichou F. and Elber G., “Output Sensitive Extraction of Silhouettes from Polygonal Geometry,” Proc. 7th Pacific Graphics Conf., IEEE CS Press, pp. 60-69, 1999.

[13] Gooch B., Sloan P.P. J., Gooch A., Shirley P. Riesenfeld R.,“Interactive Technical Illustration,”. Interactive 3D Graphics, ACM Press, 31-38, 1999

[14] Luebke D.P., “A Developer’s survey of polognal simplification algorithms. In IEEE Computer Graphics and Applications, 2001

[15] Tom H. "Silhouette Tracking", www.bytegeistsoftware.com/various/SilhouetteTracking.pdf