A Fast Silhouette Detection Algorithm for Shadow Volumes in Augmented Reality

Hoshang Kolivand, Mahyar Kolivand, Mohd Shahrizal Sunar, Mohd Azhar M. Arsad

Abstract—Real-time shadow generation in virtual environments and Augmented Reality (AR) was always a hot topic in the last three decades. Lots of calculation for shadow generation among AR needs a fast algorithm to overcome this issue and to be capable of implementing in any real-time rendering. In this paper, a silhouette detection algorithm is presented to generate shadows for AR systems. $\Delta+$algorithm is presented based on extending edges of occluders to recognize which edges are silhouettes in the case of real-time rendering. An accurate comparison between the proposed algorithm and current algorithms in silhouette detection is done to show the reduction calculation by presented algorithm. The algorithm is tested in both virtual environments and AR systems. We think that this algorithm has the potential to be a fundamental algorithm for shadow generation in all complex environments.

Keywords—Silhouette detection, shadow volumes, real-time shadows, rendering, augmented reality.

I. INTRODUCTION

Silhouette is the most important part of geometrically based shadow generation techniques. Silhouette is the only part of occluders which contributes in shadow generation. Silhouettes consist of the edges of occluders which belong to two different surfaces in which one of them can be seen from the light source position and the other one cannot. In other words, the normal vector of each surface is towards the position of light source, while the other is not. The first usage of silhouette is referred to Shadow Volumes [6] which is the fundamental research in geometrically based shadow generation techniques. The main issue of the geometrically based shadow generation techniques is silhouette detection which requires a lot of calculations. Nevertheless, improving silhouette detection is in order to enhance the frame per second and quality of shadows especially in AR [9] which will be employed in real-time. In these cases, when the occluders move or light source position changes, the silhouette will be recalculated in real-time.

Many researchers have spent time to improve shadow volumes based on silhouette detection [1], [7], [4], [10]. Fig. 1 illustrates the concept of silhouette detection. It means that the silhouette of occluders with respect to the light source is important in constructing the shadows. There are two widely used algorithms in silhouetted detection for shadow volumes. Batagelo et al. [3] proposed Hierarchical Face Clusters (HFC) algorithm and Jung et al. [7] presented BSB Tree algorithm. Both of them perform with $O(n^2)$.

Shadow volumes are implemented as a part of outdoor rendering environment to construct the precise shadow on arbitrary objects. Shadow volumes are expensive but the only reason to use these types of shadows is their accuracy without any aliasing. The theory of shadow volumes is illustrated in Fig. 2. The volume between the occluder and shadow receiver is constructed based on the silhouette of the occluders.

The widely used techniques such as BSB Tree and HFC algorithms are difficult to be improved as many years have passes and they are still in use. Thus, the techniques are appreciated. In this paper, we have presented an algorithm called $\Delta+$ Algorithm to speed up the running time for shadow volumes to be employed in real-time rendering like AR.

II. SILHOUETTE DETECTION

As mentioned early, silhouette of a blocker is the only part of object which contributes in shadow generation. Thus, to create shadows, silhouette of the occluders must be recognized first. This step needs more improvements to enhance the speed of rendering. Calculation of this detection is almost expensive and it needs to be taken into consideration for
more improvements. In this paper, we have presented an idea to reduce the calculation of this step in shadow generation geometrically and then apply it in AR to be tested in a real-time environment. To begin with, we need to analyze the fundamental concept of silhouette detection in shadow generation techniques.

Given that:

\[ E \] is the view point which in located in the camera point of view
\[ f_i \] is \( i^{th} \) face of \( F \) which is the set of all faces of the occluder
\[ n_{f_i} \] is normal vector of \( f_i \)
\[ v_f \] is a vertex of \( f \)
\[ n \] is an edge of the occluder in the set of \( E \)
\[ e_{f_i}, i^{th} \) face includes \( e \)
\[ p_f \] is a surface consists \( f \)

The distance between \( E \) and \( f \) is:

\[
\|E, f_i\| = (E - v_{f_i}) \cdot n_{f_i} 
\]

(1)

Now, if \( \|E, f_i\| \) is positive, then the face \( f_i \) is front face, otherwise it is a back face. Face recognition algorithm is presented in Algorithm 1.

Algorithm 1 Face Recognition

Step 1. If \( \|E, f_i\| > 0 \) then
\( f_i \) is a front face
Else
\( f_i \) is a back face
End if

Step 2. If \( \|E, e_{f_i} - v_f\| \cdot \|E, e_{f_i} - v\| < 0 \) then
\( e \) is a silhouette
End if

Normalization is suitable for smooth edges which is presented in (2). A simple theory of silhouette detection is provided in Algorithm 2.

\[
D_i = \frac{(f_{v_i} - v)n_{f_i}}{\|n_{f_i}\| \|f_{v_i} - v\|} 
\]

(2)

III. \( \Delta+ \)Algorithm

The regular and basic technique for silhouette detection is by going through all edges and checking one by one whether it is a contributor in shadow generation with respect to the light source or not. The algorithm is not complicated but expensive in terms of rendering. Here, we have presented an algorithm to over come this issue by optimizing the number of edges which need to be checked. The idea behind \( \Delta+ \)Algorithm comes form image based techniques [14], which compare two different view points; but, here we extend the ray from light source view point and compare to the other view point which is camera point of view.

\( \Delta+ \)Algorithm is geometrically-based and it is used to figure out the silhouette in the shorter period of time compared to the current techniques. \( \Delta+ \)Algorithm is presented in the Algorithm 3.

Algorithm 2 Silhouette Theory

Step 1. If \( D_i > 0 \) then
\( f_i \) is front face
Step 2. if \( D_i < 0 \)
\( f_i \) is back face
Step 3. \( D_i = 0 \) then
\( v_i \) is a silhouette
End if

Algorithm 3 \( \Delta+ \)Algorithm

Step 1. Calculate the number of faces and edges of occluder in advance
Step 2. For all edges
Step 3. Extend the edge by \( \epsilon \) to a new point with respect to the light source position
Step 4. Extend the Ray from light source to \( \epsilon \) which will be called extended point
Step 5. For all visible faces
Step 6. If the ray and faces have any intersection, the edge is silhouette, otherwise no

To evaluate technically, \( \Delta+ \) Algorithm needs to be explored.

Given that the light source is \( L \) and the camera point is \( C \), consider \( pq \) is a single edge in \( E \). Extend of \( p \) is shown as \( p + \Delta p \) where \( \Delta p \rightarrow 0 \). In the next step, the ray (\( R \)) which is drown from light source to the vertex of \( p \) needs to be extended to \( p' = p + \Delta p \) and then extends to infinity. If the extended ray intersects a face, \( pq \) is not a silhouette, else it is.

Algorithm 4 \( \Delta+ \) Algorithm (Details)

Step 1. For each vertex of \( pq \) in \( E \)
Step 2. \( p' = p + \Delta p \)
Step 3. Extend \( R= \)ray of light source to \( p' \)
Step 4. For all visible (\( f \)) Faces
Step 5. If Intersection(\( R, f \))=false then
Step 6. \( pq \) is a silhouette

In calculation of extend edge, there are 3 summations. To determine the intersection between extended ray and faces,
the algorithm puts the ray in the surface equation and checks whether the intersection happens or not. Therefore, given that \( x = x0 + \alpha n \) is the ray; to calculate this ray 3 Multiples and 3 Summations are needed. Given that the number of faces in the occluder is \( F \), Table I shows the complexity of \( \Delta^+ \) Algorithm. Then Complexity = \( O(n \log_2 n) \).

### IV. \( \Delta^+ \) Algorithm in Shadow Volumes

Here, we have created shadow volumes using stencil buffer. Silhouette detection is the main part of this algorithm. \( \Delta^+ \) Algorithm recognizes the silhouette as can be seen in Fig. 3.

To use \( \Delta^+ \) Algorithm in shadow volumes, the first step is creating the volume between occluder and shadow receivers. Then, the rays from light source to the vertex are given and extends to the shadow receivers are calculated. The regular process of shadow volume is applied. Meanwhile, when the ray inters into the front face, the stencil buffer is increased, when the ray exits form the back of the face the stencil buffer is decreased. At the end of this process, the amount of stencil buffer is checked. If the stencil buffer is zero the point is in lit else in shadows. The algorithm summarized in Algorithm 5.

If the camera point is out of the shadow region Z-pass algorithm is employed, while, Z-fail algorithm is used when it is inside the shadow region. In fact, steps 5 and 6 will be changed with following steps:

#### Algorithm 5 Shadow volume generation using \( \Delta^+ \) Algorithm

1. **Step 1.** Render the whole scene when light is off
2. **Step 2.** Disable depth buffer
3. **Step 3.** Enable stencil buffer
4. **Step 4.** \( \sum_{i=1}^{Np} \sum_{j=1}^{NP} II((not(intersection(R_e,f)), Add to S, Nil)) \)
5. **Step 5.** \( \sum_{i=1}^{NP} II((P_i, R_v), \sum_{j=1}^{NP} II(ZTest, Stencil +, Nil), Nil) \)
6. **Step 6.** \( \sum_{i=1}^{NP} II((Not(P_i), R_v), \sum_{j=1}^{NP} II(ZTest, Stencil −, Nil), Nil) \)
7. **Step 7.** Render scene again with lighting
8. **Step 8.** Enable to write in color buffer
9. **Step 9.** If (Stencil mod 2=0) then Keep Stencil

\( NE \) : Edge number
\( NF \) : Face number
\( NP \) : Polygon number
\( S \) : Array to keep Silhouette edges

<table>
<thead>
<tr>
<th>Line</th>
<th>Worst case</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>For all vertices of ( pq ) in ( E )</td>
</tr>
<tr>
<td>2.</td>
<td>( P = p + \Delta p )</td>
</tr>
<tr>
<td>3.</td>
<td>( R = Ray ) of light source to ( p )</td>
</tr>
<tr>
<td>4.</td>
<td>( R = Ray ) of light source to ( p )</td>
</tr>
<tr>
<td>5.</td>
<td>( R ) if ( intersection(R_e,F) ) is inside the shadow region. In fact, steps 5 and 6 will be changed with following steps:</td>
</tr>
<tr>
<td>6.</td>
<td>( pq ) is a silhouette</td>
</tr>
</tbody>
</table>

### Table I Complexity of Algorithm

<table>
<thead>
<tr>
<th>Line</th>
<th>Worst case</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>For all vertices of ( pq ) in ( E )</td>
</tr>
<tr>
<td>2.</td>
<td>( P = p + \Delta p )</td>
</tr>
<tr>
<td>3.</td>
<td>( R = Ray ) of light source to ( p )</td>
</tr>
<tr>
<td>4.</td>
<td>( R = Ray ) of light source to ( p )</td>
</tr>
<tr>
<td>5.</td>
<td>( R ) if ( intersection(R_e,F) ) is inside the shadow region. In fact, steps 5 and 6 will be changed with following steps:</td>
</tr>
<tr>
<td>6.</td>
<td>( pq ) is a silhouette</td>
</tr>
</tbody>
</table>

#### Fig. 3 Illustration of stencil buffer in shadow volumes taking silhouette detection into account

#### Fig. 4 Shadow volumes using \( \Delta^+ \) Algorithm

#### Table I Complexity of Algorithm

<table>
<thead>
<tr>
<th>Line</th>
<th>Worst case</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>For all vertices of ( pq ) in ( E )</td>
</tr>
<tr>
<td>2.</td>
<td>( P = p + \Delta p )</td>
</tr>
<tr>
<td>3.</td>
<td>( R = Ray ) of light source to ( p )</td>
</tr>
<tr>
<td>4.</td>
<td>( R = Ray ) of light source to ( p )</td>
</tr>
<tr>
<td>5.</td>
<td>( R ) if ( intersection(R_e,F) ) is inside the shadow region. In fact, steps 5 and 6 will be changed with following steps:</td>
</tr>
<tr>
<td>6.</td>
<td>( pq ) is a silhouette</td>
</tr>
</tbody>
</table>
Fig. 5 View 1: Result of Δ+ algorithm in AR

Fig. 6 View 2: Result of Δ+ algorithm in AR

detection [5], [13]. It should be mentioned that these two algorithms are currently widely used in silhouette detection due to the fundamental concepts. Table II shows a simple comparison between the complexity of HFC, BSB Tree and Δ+ Algorithm in the case of complexity of the algorithms.

<table>
<thead>
<tr>
<th>Number of Vertices</th>
<th>Algorithms</th>
<th>HFC</th>
<th>BSB Tree</th>
<th>Δ+ Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>10000000</td>
<td>1200000</td>
<td>10000</td>
<td></td>
</tr>
<tr>
<td>5000</td>
<td>23000000</td>
<td>26000000</td>
<td>60000</td>
<td></td>
</tr>
<tr>
<td>20000</td>
<td>3800000000</td>
<td>3840000000</td>
<td>300000</td>
<td></td>
</tr>
<tr>
<td>100000</td>
<td>870000000000</td>
<td>123E+11</td>
<td>17000000</td>
<td></td>
</tr>
</tbody>
</table>

TABLE II
COMPARISON BETWEEN THE COMPLEXITY OF HFC [7], BSB TREE [3] AND Δ+ ALGORITHM

V. AR SYSTEMS

A virtual object set within a real environment constitutes an AR system [2], [9], [8]. An AR-system incorporates more real objects and a few virtual objects with the real AR taking a dominant role over the virtual. Nowadays, many researchers have focused on AR since most of the computer graphics applications require computer-generated objects to be seamlessly integrated into natural images or videos such as environmental assessments and computer games. Moreover, the appearance of virtual objects should reveal the consistence with effect of the interaction between objects or even sky color in outdoor rendering.

Shadows in AR need sophisticated work. Huge amount of calculation for uploading 3D objects in AR systems demands a fast shadow generation technique which Δ+ Algorithm in shadow volumes covers this requirement. The main issue in realistic virtual objects in virtual environments and AR system is remaining the balance between realism and FPS. Increasing the realistic is not a big issue if the speed of rendering is not taken into account. Noh et al. [12] tried to generate the realistic soft shadows in AR adding soft projection shadows. Madsen et al. [11] considered the sky illumination and volume shadows in AR to enhance the realism.

Figs. 5 and 6 show implementation of the Δ+ algorithm in AR. The shadows are accurate enough and no so heavy in rendering.

The AR system including shadow volumes using Δ+ Algorithm is performed in 74.35 FPS. The 3D objects include 836 faces. In fact, the results show that the presented algorithm is appropriate to be used in any real-time rendering.

VI. CONCLUSION

Silhouette detection is the heaviest part of geometrically-based algorithms for shadow generation. In this paper, we have presented Δ+ Algorithm to overcome the heaviest calculation of silhouette detection. A simple idea which came from image based techniques to reduce the number of edges which contribute in shadow generation are taken into account. Δ+ algorithm is working by extending each ray from light source to visible vertices of occluders then using a simple checking can recognize whether the edge is silhouette or not. The algorithm is evaluated by comparing with two other widely used fundamental algorithms. The efficiency of Δ+ algorithm is $O(n \log_2{n})$ compared to the others which are $O(n^2)$. Employing Δ+ algorithm on shadow volumes creates a fast enough shadow for real-time rendering. Finally, shadow volumes using Δ+ algorithm are employed on AR to test the algorithm in real-time environments resulting an accurate and fast shadow generation in AR.

ACKNOWLEDGMENT

This research was supported by Vot. Q.J130000.2528.12H18 RUG grant at the Magic-X, Universiti Teknologi Malaysia.

REFERENCES


Hoshang Kolivand received his MS degree in applied mathematics and computer from Amirkabir University, Iran, in 1999, and his PhD from Media and Games Innovation Centre of Excellence (MaGIC-X) in Universiti Teknologi Malaysia. Previously, he worked as a lecturer in Shahid Beheshti University, Iran. He has published numerous articles in international journals, conference proceedings and technical papers, including chapters in books. He is an active reviewer of many conference and international journals. He has also published many books in object-oriented programming and mathematics. His research interests include computer graphics and Augmented Reality.

Mahyar Kolivand is currently a year 11 student at Seri Omega International School and a research member of MaGIC-X at Universiti Teknologi Malaysia, Skudai. His research focus is on game developing, computer vision, computer graphics, and medicine.