Master’s Thesis

Single-Pass Stereo Rendering with Bidirectional Image Warping

Jaemyung Kim

Department of Platform Software
The Graduate School
Sungkyunkwan University
Master’s Thesis

Single-Pass Stereo Rendering with Bidirectional Image Warping

Jaemyung Kim

Department of Platform Software
The Graduate School
Sungkyunkwan University
Single-Pass Stereo Rendering with Bidirectional Image Warping

Jaemyung Kim

A Master’s Thesis Submitted to the Department of Platform Software and the Graduate School of Sungkyunkwan University in partial fulfillment of the requirements for the degree of Master of Science in Engineering

October 2019

Approved by Sungkil Lee

Major Advisor
This certifies that the Master’s Thesis of Jaemyung Kim is approved.

__________________________________________________________________________

Major Advisor

__________________________________________________________________________

Committee Chair

__________________________________________________________________________

Committee Member

The Graduate School
Sungkyunkwan University
December 2019
## Contents

- List of Tables .......................................................... ii
- List of Figures .......................................................... ii
- Abstract ................................................................. iii
- 1. Introduction .......................................................... 1
- 2. Related Work ........................................................ 4
- 3. Algorithm .............................................................. 7
  - 3.1 Geometry Shader based Geometry Redirection ............... 9
  - 3.2 Visibility Comparison ............................................ 12
    - 3.2.1 Frustum Culling based Visibility Comparison .......... 12
    - 3.2.2 Back Face Culling based Visibility Comparison ....... 15
  - 3.3 Warping and Stereo Image Composition ....................... 18
    - 3.3.1 Motion Texture Generation ................................ 18
    - 3.3.2 Shared Memory based One-Dimensional Warping ....... 20
      - 3.3.3 Crack Removal ............................................. 24
- 4. Results ............................................................. 26
- 5. Conclusion and Discussion ........................................ 33
- References ................................................................ 36
- Korean Abstract ......................................................... 40
List of Tables

Table 1. ................................................................. 22
Table 2. ................................................................. 26
Table 3. ................................................................. 31

List of Figures

Fig.1. ................................................................. 8
Fig.2. ................................................................. 10
Fig.3. ................................................................. 11
Fig.4. ................................................................. 13
Fig.5. ................................................................. 14
Fig.6. ................................................................. 16
Fig.7. ................................................................. 17
Fig.8. ................................................................. 19
Fig.9. ................................................................. 21
Fig.10. ................................................................. 23
Fig.11. ................................................................. 25
Fig.12. ................................................................. 27
Fig.13. ................................................................. 32
Fig.14. ................................................................. 35
Abstract

Single-Pass Stereo Rendering with Bidirectional Image Warping

Stereo rendering is a method of imparting a three-dimensional effect to image contents such as a movie or a game. However, there is a problem in that the computational load is increased by rendering for both eyes. Stereo rendering using image warping can reduce geometry computation of both eyes into once, so it guarantee high performance when rendering complex geometries. However, the hole caused by the difference between the stereo images causes quality and performance degradation. In this paper, we propose a single-pass stereo rendering technique using bidirectional warping to minimize the occurrence of hole. The proposed technique suppress the occurrence of hole by increase the sampling area of the geometry. Geometries divided and rendered to both eye based on visibility. Then they warping in both directions to generate complete stereo image. This technique shows high stability compared to the conventional image warping based stereo rendering technique even when stereo rendering with low visual similarity of both eyes. It can be effectively used for renderings requiring complex geometric and shading calculations.

Keywords: virtual reality, stereo rendering, image warping, GPU
I. Introduction

Stereoscopic rendering requires rendering a scene twice for each eye, which inherent involves generation of binocular views. Straightforward and most accurate approach is simply rendering the geometries twice, but is often costly. Not only does the calculation double, but it also requires a new pass through the rendering pipeline, which incurs additional overhead in transferring geometry from the CPU to the GPU once again.

Therefore, many approaches had proposed for create two images using a single rendering pass. NVIDIA has made this possible through hardware-based approaches on its latest GPUs [1], but there is also problems: hardware dependencies and after geometry processing still require twice as much computation as before.

An alternative software based approach is generating another view from the one rendered scene without geometry processing but with warping. This approach can benefit from complex scenes, when the warping is very efficient. Warping is a technique that reuses rasterized geometry with transform, replacing geometric vertex processing with pixel operations. In modern graphics environment, geometric complexity is extremely higher than pixel complexity. The performance improvement depends on the efficiency and resolution of the warping, generally the performance improvement is over 50% in complex scenes.
Previously, warping has been used for many purposes, including view morphing/interpolation [20, 21, 25–29], multi view generation for distributed effects such as motion/lens blur [33], depth–of–field (DOF) [17, 18], and soft shadows [22, 23]. Stereoscopic rendering can be considered a limited case of such scenarios, generating only a single another view.

Major problem of the stereoscopic warping is efficiency and hole filling, because a single view always misses hidden surfaces occluded by the foreground visible objects. Hidden surfaces in one view may be visible surfaces in another view, but warping cannot create pixels cause it is reuse technique. So when an unsampled hidden surface is revealed, this part is left empty, which is called a hole.

To handle this problem, Didyk et al. [2] proposed a method of inpainting hole using the image of the previous frame. Schollmeyer et al. [3] improved the performance and quality by using low–level images to filling hole, and Oculus’s re–rendering of hole yielded very high quality [4]. Bowles et al. [5] showed that approximation of the original pixel can reduce the small hole.

Forward warping can be accurate, when the source pixels are sampled enough. But in case of insufficient sampling, holes might occur in the warped view. Backward warping [5] can handle much better than the forward warping without grid–based finite–area warping (or the expansion of the support).

One motivating study is DOF synthesis from sparse views [6], which
samples geometry from the multiple views. This approach synthesize novel views from multiple views. Our approach is similar, but we distribute the geometries to multiple (precisely, two views for stereoscopy). Thereby, a single geometry is processed mostly once.

In this paper, we present the concept of a bidirectional warping technique that eliminates the potential hole so that no additional complex process is required. Then we also present implemented single-pass stereo rendering technique with bidirectional warping.

Our stereo rendering technique consists of geometry shader based geometry redirection and shared memory based one-dimensional warping. The technique can be effectively used for stereo rendering, which requires complex geometric and shadow calculations, and can be extended to multi-view rendering.

Precisely, our contributions include:
- Bidirectional scene decomposition pipeline that minimizes potential holes
- Pipeline implementation based on geometry shader and forward warping
- Atomic forward warping technique optimized for stereo rendering
II. Related Work

Single-pass stereo rendering techniques that generate stereo images in one rendering can be categorized into hardware-based and software-based software based on image warping.

In the hardware-based case, the goal is to create two images in one rendering pass. Single Pass Stereo Rendering [1], supported on NVIDIA GPUs, made it possible to render two images at once by setting a new parameter to position in the second view as the output of the vertex shaders, geometry shaders, and tessellation shaders that handle geometry. Accurate stereo image generation is possible, but requires specific hardware and the computation after geometry processing is equally doubled.

The software-based stereo rendering technique mainly aims at generating an additional image by warping the shaded and post-processed images. They can be classified into forward warping and backward warping.

Forward warping creates an image by moving the original pixel to a new location. Didyk et al. [2] have shown that possible efficiently render complex geometries by using tessellation–warping and altering the rendered field of view every frame, then filling hole with warped frame from the previous frame. However, the stereo image itself was a one-dimensional shift, it required
complex two-dimensional warping to get the pixels from the previous frame, and in the process of changing the field of view rendered every frame caused visual artifacts such as disturbing reflection light. Schollmeyer et al. [3] used low-level images and ray casting techniques to filling hole. Although the image quality was improved through this, processing for filling hole still took more time than warping.

Backward warping searches for the original pixel at the new location to create an image. Bowles et al. [5] have shown that the approximation of the original pixels using fixed-point iteration can reduce the generation of hole, but if the area of the sampled pixels is large because of the dissimilarity of the binocular field of view, the additional hole still remain.

Thus, in software-based stereo rendering, processing of hole generated during warping has been a major issue [30, 31]. Therefore, this paper aims to minimize the hole creation during warping to reduce the weight of additional computation.

Conventional techniques focused on warping the whole image created in one view to another view, but we divides the geometry according to visibility and generates two partial images and then warps them. This increases the number of pixels sampled as much as possible to reduce the potential for creating hole.

The basic idea is drawn from the research of Yang et al. [6]. They have shown that warping two consecutive frames in both directions can produce a high quality intermediate image. This study transforms it to fit stereo rendering
so that a high quality stereo image can be generated by dividing one frame into two and warping them in both directions.
III. Algorithm

Bidirectional warping consists of geometry shader-based geometry redirection and warping to produce the final stereo image. This chapter describes the geometry redirection phase, the comparison of visibility performed during redirection, and the specific description of the warping and final image generation phase.

Figure 1 shows an overview of the algorithm. Each primitive in the geometry is redirected and rendered to a buffer corresponding to each view according to visibility. Then, the rendered image is combined with the warped image in the opposite view to produce a stereo image.

This chapter provides a detailed description of the geometry redirection phase, warping and combination phases described in the overview picture, and how it was implemented.
Fig. 1. Overview of rendering process
1. Geometry Shader based Geometry Redirection

While passing through the geometry shader, each primitive goes through a visibility comparison for each field of view. Then they redirected to the layer with the highest visibility. In the worst case, primitive redirected to both fields of view.

The fundamental reason for holes in image warping is that color can’t be determined if pixels are not sampled in the source image. Figure 2 shows the difference in the possibility of hole creation according to the warping direction. If the sample of the original image is smaller than what is required for the target image, as in (a), no matching pixel can be found, resulting in a hole. Warping with fewer samples, such as (b), does not generate holes because all the pixels needed are matched.

This process ensures that primitives are rendered in the most visible state and minimizes the creation of potential hole by increasing the pixels sampled. This require a way to correctly compare the visibility of primitives in each field of view, as described in next section.
Fig. 2. Hole creation along sampling area. Because sampled pixels in primitives are finite, holes can occur depending on the direction of warping. (a) If warping destination image require more pixels than sampled, holes will occur. (b) No holes are created in the opposite direction.
Fig. 3. Redirected geometry based on visibility. (a) left eye, (b) right eye rendered texture
2. Visibility Comparison

Visual surface determination (culling), which determines whether a geometry is rendered, is one of the oldest research topics in the field of computer graphics for performance optimization, and a number of techniques have already been studied [7]. By transforming them into inter-view comparison, they can be used to compare the visibility of the proposed technique.

In this paper, we used frustum culling and back-face culling for our application to implement stereo rendering, they are two of the most popular surface decision algorithms. In addition to the two algorithms used, it can be extended with various culling techniques, such as occlusion culling and contribution culling.

2.1 Frustum Culling based Visibility Comparison

As shown in Figure 4, depending on the positional relationship between the primitive and the view frustum, the primitive may not be rendered or may be rendered in a truncated shape. Thus, in order to compare whether primitives are rendered intact in each view, how much each primitives contained within the frustum can defined as visibility, which can be calculated using position in the screen coordinate system.
Define primitives as $G$, views as $V$, and the visibility function $f_r(G, V)$ as follows: $G_{x_{\text{max}}}^V$ is the x-coordinate of the vertex with the largest absolute value of the x-coordinate of the three vertices of the primitive $G$ in the screen space of the view $V$.

$$f_r(G, V) = 1/\max(1, |G_{x_{\text{max}}}^V|) \quad (1)$$

The value of $f_r(G, V)$ is fixed to 1 if there are primitives in the view frustum and decreases if it leaves the frustum. Figure 5 show implementation of $f_r(G, V)$ in geometry shader.
Algorithm 1: Shader code of getting visibility based on frustum culling.

```c
float l_max = 1, r_max = 1;

// calculate distance from center per vertex
for(int k = 0, kn = gl_in.length(); k<kn; k++){
    vec3 pos = gl_in[k].gl_Position.xyz;
    // transform to clip space position
    vec3 lepos = (lcam.view_matrix*vec4(pos, 1)).xyz;
    vec4 lcpos = lcam.projection_matrix*vec4(lepos,1);
    lcpos /= lcpos.w;
    vec3 repos = (rcam.view_matrix*vec4(pos, 1)).xyz;
    vec4 rcpos = rcam.projection_matrix*vec4(repos,1);
    rcpos /= rcpos.w;

    // get maximum distance
    r_max = max(r_max, abs(rcpos.x));
    l_max = max(l_max, abs(lcpos.x));
}

// get visibility
float r_vis = 1/r_max;
float l_vis = 1/l_max;
```

Fig. 5. shader code of getting visibility in geometry shader

Then compares the computed result to determine which view the primitive will be redirected to. If both views have the same visibility, additional visibility comparisons based on back-face culling as described in next section determine which views will be redirected.
2.2 Back Face Culling based Visibility Comparison

The following equation represents one of the common implementations of back face culling in the world coordinate system [8].

\[
\text{Discard when } (\mathbf{p}_{\text{cam}} - \mathbf{p}_{\text{primitive}}) \cdot \mathbf{n}_{\text{primitive}} \quad (2)
\]

In this equation, \( \mathbf{p} \) is the position coordinate in the world coordinate system, \( \mathbf{n} \) is the surface normal vector, and using the dot product of the vector to calculate the angle between the surface normal vector and the camera-primitive vector in the world coordinate system.

The calculated angle is used to determine if the surface direction of the primitive is away from the camera (backface), but it can also be used for comparison of the projected areas.

Figure 6 shows the correlation between the angle formed by the normal vector of the camera and the primitive and the projection area.
The smaller the calculated angle, the larger the projection area becomes as the primitive faces the camera. Therefore, assuming visibility as the area of the projection area, the visibility function $f_b(G, V)$ can be defined as follows using Equation (2).

$$f_b(G, V) = |(p_V - p_G) \cdot n_G| \quad (3)$$

If program doing backface culling already, remove the absolute value entry and use the following formula.

$$f_b(G, V) = (p_V - p_G) \cdot n_G \quad (4)$$
Figure 7 show implementation of $f_{o}(G; V)$ in geometry shader. This is used to compare the visibility in each view to determine the final redirection direction of the primitive, and if the visibility at this stage is the same, redirect to both (worst case).

**Algorithm 2: Shader code of getting visibility based on backface culling.**

```c
// calculate surface normal
vec3 a = (gl_in[1].gl_Position-gl_in[0].gl_Position).xyz;
vec3 b = (gl_in[2].gl_Position-gl_in[0].gl_Position).xyz;
vec3 norm = normalize(cross(a,b));

// find center position in the triangle
vec3 pos = vec3(0);
for(int k=0, kn=gl_in.length(); k<kn; k++)
    pos+=gl_in[k].gl_Position.xyz;
pos /= float(gl_in.length());

// get visibility
vec3 el = normalize(lcam.eye-pos);
vec3 er = normalize(rcam.eye-pos);

float l_vis = abs(dot(el,norm));
float r_vis = abs(dot(er,norm));
```

Fig. 7. shader code of getting visibility in geometry shader
3. Warping and Stereo Image Composition

The image rendered in each field of view is warped to the opposite field of view and combined with the image of the opposite field of view to produce the final stereo image. This section describes the final stereo image composition method in detail and proposes an implementation using fast shared GPU memory to improve the execution speed of one-dimensional forward warping.

3.1 Motion Texture Generation

To warp an image, calculations are required of how each pixel should move. To get this, reproject the position stored in g-buffer to the view to be warped. This process is carried out in the following order. First, the view space coordinates stored in the g-buffer are reconstructed into positions in world space using the inverse view matrix of the source camera. It is then projected into the screen space of the destination camera using the view camera’s view and projection matrix.

The reprojected depth and difference between the reprojected screen space coordinates and the original screen space coordinates is then stored as a texture. The texture is then subjected to minmax mipmap filtering to determine the maximum/minimum of motion, which is used for workgroup placement as described in 3.2.
Fig. 8. Rendered texture and generated motion textures for (a) left eye and (b) right eye. The motion texture has two channels of distance in screen space and a reprojected depth.
3.2 Shared Memory based One-Dimensional Warping

Forward warping is a very simple technique that theoretically ends just by moving the original pixel to a new location. However, since there is a possibility that a plurality of pixels can be mapped to one position, when a plurality of pixels are mapped, it should be processed to display only the pixel value having the lowest depth. If this is not done, noise will be generated in the area due to the parallel nature of the GPU.

In the recent forward warping techniques [8, 9, 10], the depth value per pixel is first obtained by only warping the depth value, and then only the pixel corresponding to the minimum value is processed.

Forward warping is implemented using compute shader and atomic operations. Compute shader is used to compute the locations to be warped in parallel, and computing the minimum depth that maps to that pixel with atomicMin operations and SSBO.

Because warping of depth values is done in parallel on the GPU, synchronization between threads is required until the minimum is determined. In the conventional technique, for this purpose, the rendering path was divided into two to perform synchronization on the entire image. However, this causes additional CPU-GPU data exchange.

In this paper, we propose a shared memory based method that optimizes this
for stereo rendering. In stereo rendering, only pixels that are likely to be mapped to the same pixel are located in the same row, so synchronization is sufficient only in rows.

As shown in Figure 9, we set the compute shader's workgroup to correspond to a row in the image to synchronize within the row pixels. Grouping them into the same workgroup allows synchronization and warping within a single rendering path without having to divide the rendering path. Furthermore, shared memory with very fast access can be used.

![Diagram](Image)

**Fig. 9.** Existing method (left) and proposed method (right). Performance increased using shared memory instead of global memory.

The detailed process is as follows. Bind one-dimensional textures of the same width and size to the shared memory, and then warp only depth values to find the minimum depth of the mapped pixels. At this time, the image of the opposite field of view is also included in the minimum calculation to
simultaneously process the combination with the image of the opposite field of view. After this, the workgroup is synchronized and the color value of the pixel corresponding to the minimum depth of the mapped pixels is shifted.

Unfortunately, because the work group has a size limit, we couldn’t assign every pixel of the image row to the work group. Therefore, we had place several workgroups in a row as shown in Figure 10. They overlapped the area shared by the workgroups by the max value of motion for avoid artifacts. There is some computational loss, but still no require synchronization between workgroups.

As shown as Table 1, the forward warping of the proposed scheme reduces execution time compared to the conventional method [8].

<table>
<thead>
<tr>
<th></th>
<th>Ours</th>
<th>Cichocki 2017 [8]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance (ms)</td>
<td>0.653</td>
<td>0.766</td>
</tr>
</tbody>
</table>
Fig. 10. Work group setting for implementation. Two adjoining workgroups share and process as many pixels as the maximum distance of motion.
3.2 Crack Removal

Due to the error of floating point calculations in both views, after the movement, a fine crack as shown in Figure 11 occurs.

Since our technique is based on forward warping, we cannot find holes directly in the resulting image. Primitive, but we checking every pixel in the texture of the rendering scene to find hole when stereo warping has been finished. Based on the fact that the depth of the hole is larger than the depth of the normal pixel, determining hole to be inpainting.

Inpainting is simple filtering, if there are empty pixels, remove them by interpolating with the pixels next to them. Since the crack is very small about 1px, it is hard to see with the naked eye even if some errors occur during interpolation, so simple linear interpolation is used.
Fig. 11. Results of crack removal with linear interpolation
IV. Results

The proposed stereo rendering using two one-dimensional warping shows significantly improved quality compared to the stereo rendering techniques using conventional warping.

For quality measurement, the quality of each technique was measured by changing the distance with focal plane (DF) in the same scene. Figure 12 shows the results generated using the reference (2-Pass stereo) and the proposed method and the existing warping method. Table 2 shows the results of measuring the quality of each technique compared to the reference using SSIM and PSNR [11]. BW means backward warping and FW means forward warping, respectively. The distance between the two fields of view (IPD, Interpupillary Distance) was set to 64mm.

Table 2. Quality comparison results between each technique (SSIM/PSNR)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6000</td>
<td>0.994/38.6</td>
<td>0.962/22.6</td>
<td>0.967/23.2</td>
</tr>
<tr>
<td>2000</td>
<td>0.998/43.8</td>
<td>0.953/20.6</td>
<td>0.958/21.0</td>
</tr>
<tr>
<td>300</td>
<td>0.995/39.5</td>
<td>0.779/12.9</td>
<td>0.781/12.9</td>
</tr>
</tbody>
</table>
Fig. 12. Warped image of each technique (focal distance: 6000, scene: Crytek Sponza)
Fig. 12. Warped image of each technique (focal distance: 2000, scene: Crytek Sponza)
Fig. 12. Warped image of each technique (focal distance: 300, scene: Crytek Sponza)
Proposed method show overall better quality than the existing technique. For images with low visual similarity, it can be seen that the quality is particularly higher than that of the conventional technique.

In addition, the performance of each technique was measured while varying the complexity of the scene to verify the effectiveness of the technique. Table 3 shows the execution time of each measured technique divided into rendering and warping. The complexity of the scene was measured based on the number of triangle primitives used.

Two warping processes significantly increase the time spent on warping compared to conventional techniques, and the time spent on the rendering itself also increases. Proposed stereo rendering method resulting in the lowest performance among the compared techniques at relatively low complexity.

However, higher the geometric complexity of the scene, our stereo rendering method getting higher the performance advantage compared to simple 2-pass stereo rendering. The increase in execution time during rendering seems to be due to the load occurred through the geometry shader during rendering. Especially when rendering a scene of 10M, the performance loss is noticeable, and is slower than the 2-pass rendering. A geometry shader configured for test to pass through the geometry without processing showed similar performance loss. Geometry shaders have an additional primitive assembly stage for primitive processing, in which the vertex collection and alignment are performed. In this process, over-computation seems to be performed. The higher the number of
vertices compared to the number of primitives, the lower the performance was observed. The 10M scene seems to have a worse performance penalty because it has a higher percentage of vertices than other scenes.

Table 3. Performance comparison results (ms) of each technique (image resolution: FHD, GPU: NVIDIA GeForce GTX 1080)

<table>
<thead>
<tr>
<th>Complexity</th>
<th>Method</th>
<th>Rendering</th>
<th>Warping</th>
</tr>
</thead>
<tbody>
<tr>
<td>1M tri. 0.8M vert.</td>
<td>Ours</td>
<td>3.320</td>
<td>1.057</td>
</tr>
<tr>
<td></td>
<td>BW[5]</td>
<td>1.889</td>
<td>0.407</td>
</tr>
<tr>
<td></td>
<td>FW[8]</td>
<td>1.887</td>
<td>0.636</td>
</tr>
<tr>
<td></td>
<td>2-Pass</td>
<td>3.599</td>
<td></td>
</tr>
<tr>
<td>10M tri. 12M vert.</td>
<td>Ours</td>
<td>13.957</td>
<td>1.169</td>
</tr>
<tr>
<td></td>
<td>BW[5]</td>
<td>6.102</td>
<td>0.635</td>
</tr>
<tr>
<td></td>
<td>FW[8]</td>
<td>6.111</td>
<td>0.765</td>
</tr>
<tr>
<td></td>
<td>2-Pass</td>
<td>11.958</td>
<td></td>
</tr>
<tr>
<td>100M tri. 70M vert.</td>
<td>Ours</td>
<td>70.093</td>
<td>1.170</td>
</tr>
<tr>
<td></td>
<td>BW[5]</td>
<td>54.056</td>
<td>0.622</td>
</tr>
<tr>
<td></td>
<td>FW[8]</td>
<td>53.842</td>
<td>0.776</td>
</tr>
<tr>
<td></td>
<td>2-Pass</td>
<td>108.926</td>
<td></td>
</tr>
<tr>
<td>1000M tri. 600M vert.</td>
<td>Ours</td>
<td>604.827</td>
<td>1.161</td>
</tr>
<tr>
<td></td>
<td>BW[5]</td>
<td>554.255</td>
<td>0.632</td>
</tr>
<tr>
<td></td>
<td>FW[8]</td>
<td>553.643</td>
<td>0.785</td>
</tr>
<tr>
<td></td>
<td>2-Pass</td>
<td>1104.867</td>
<td></td>
</tr>
</tbody>
</table>
Fig. 13. Various rendered scene created using bidirectional warping.
V. Conclusion and Discussion

In this paper, we propose a technique to generate stereo images with low hole occurrence by two one-dimensional warping. Compared to the existing technique of warping an image in one view, the geometry is divided into two directions according to visibility, greatly reducing the generation of hole that required additional filling process, and accelerating performance by using shared memory-based forward warping optimized for one-dimensional warping.

Stereo images generated using this technique showed higher quality than conventional techniques even in scenes with less similarity of stereo images, and also achieved higher performance than simple stereo rendering in high complexity scenes. However, in the rendering process by dividing the geometry, it was observed that the execution time was greatly increased while through the geometry shader. Therefore, research to find a more efficient separation method should be conducted. Pixel level separation using hardware-based single-pass rendering will be conducted.

Our technique uses forward warping, which is a bit primitive and less fit for the GPU. Because backward warping that assumes the pixels as continuous function is difficult to fit to our method. In our method, pixels are not continuous functions because they are divided into geometry redirection. Combining the backward warping is will be a mainly issue in future research.
In addition, object occlusion is not currently considered in the visibility test process for geometry separation, which may result in quality degradation as shown in Figure 14. In order to solve this problem, a method of detecting occlusion in advance using quad-tree may be used when heavy post-processing is required. Alternatively, if the aforementioned pixel level separation is performed, more accurate separation may be performed by executing z-test.

We presented an effective, stable and scalable warping method for single pass stereo rendering. Separation of geometry based on visibility can significantly reduce the hidden surface problem of existing warping method. The quality can be further improved by applying another visibility determination methods. Our model can serve as a basis for interactive stereo rendering applications which require high scene complexity and costly shading/post-processing.
Fig. 14 Artifacts caused by the absence of visibility calculations due to occlusion. It can be solved by including occlusion culling using z-buffer or quad-tree.
References


논문요약

양방향 이미지 와핑을 이용한 단일 패스 스테레오レン더링

김재명
소프트웨어플랫폼학과
성균관대학교

스테레오 렌더링은 양안의 이미지를 다르게 함으로써 영화, 게임 등과 같은 2D 이미지에 입체감을 부여하는 방법이지만, 양안에 대한 렌더링으로 연산량 부하가 증가하는 문제점이 있다. 이미지 와핑을 이용한 스테레오 렌더링은 양안에 대한 기하 연산을 한 번으로 줄일 수 있어 복잡한 기하를 렌더링할 때 높은 성능을 가진다. 양안 이미지 간의 차이에서 발생하는 빈 공간 및 그 처리가 품질과 성능의 하락을 야기한다. 본 논문에서는 빈 공간 자체의 발생을 낮추는 양방향 와핑 기법과 이를 이용한 스테레오 렌더링 기법을 제시한다. 제안하는 기법은 기하를 가시성에 따라 양안에 나누어 렌더링하고 이를 양방향으로 와핑하는 것으로써 기하의 샘플링 면적을 넓히고, 이를 통해 빈 공간의 발생을 억제한다. 본 기법은 양안의 시각적 유사도가 낮은 스테레오 렌더링 시에도 기존 이미지 와핑 기법 대비 높은 안정성을 보이며, 복잡한 기하 및 음영 계산이 필요한 렌더링에 효과적으로 사용될 수 있다.

주제어: 가상현실, 스테레오 렌더링, 이미지 와핑, GPU
Master's Thesis

Single-Pass Stereo Rendering with Bidirectional Image Warping

Jaemyung Kim

2020