20th International Congress on Modelling and Simulation, Adelaide, Australia, 1–6 December 2013
www.mssanz.org.au/modsim2013

Ray tracing based fast refraction method for an object seen through a cylindrical glass

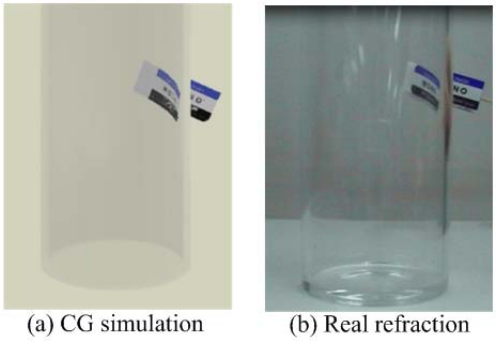
Mukai N. [a], **Y. Makino** [a] **and Y. Chang** [b]

[a] *Graduate School of Engineering, Tokyo City University, Japan*
[b] *Knowledge Engineering, Tokyo City University, Japan*
Email: mukai@cs.tcu.ac.jp

Abstract: In computer graphics, generating realistic images based on physical phenomena is one of the most challenging issues, especially for optical related phenomena such as reflection and refraction. In order to generate such images, there are two major methods: ray tracing and environment mapping. Ray tracing can generate fine images since it traces optical path for each pixel constructing the image, although it takes huge time as the resolution of the image improves and the number and complexity of the objects in the scene increases. On the other hand, environment mapping is based on the images that are previously generated so that it can generate realistic images in real time. Both methods have merits and demerits. Then, many reflection and refraction methods have been developed by utilizing the merits. In general, these techniques are designed to be used for any objects that are constructed with any numbers of polygons. On the other hand, we can calculate image distortion caused by reflection or refraction if the shape of the reflective object or transparent material is geometrically defined. Therefore, this paper proposes a new technique that calculates the positions of vertices constructing display objects, which are deformed by being seen through a transparent material. The refraction calculation is accurate because it is performed with the same method as ray tracing. In addition, the proposed method is faster than ray tracing since the calculation is performed for the vertices that construct the display objects instead of pixels that construct the final image because the number of the pixels is huge compared to that of the vertices.

We can calculate the movement vectors of vertices that construct display objects by using Snell's law if the transparent material is geometrically defined. In the case of parallel glass, refraction occurs twice because ray of light is refracted when it enters and leaves the glass. On the other hand, in the case of cylindrical glass, there are two types of refractions. Refraction occurs four times if ray passes two curved planes, and refraction occurs twice if it passes the verge of the glass. In addition, no refraction occurs if ray does not pass the transparent material. Then, three types of objects that are distorted or not distorted according to the refraction times should be generated, and the pixels constructing the final image should be indexed according to the refraction time. For indexing of the pixel, depth peeling method can be used, which detects for each pixel how many times of peeling are required to remove the transparent material. Finally, the refracted image is generated by combining the three different types of images pixel by pixel.

Figure 1 is one of applications by using the proposed method and it shows the comparison between the real image and CG simulation. The display object is an eraser and the transparent material is a cylindrical glass. Refraction occurs four times in the inside of the glass, while no refraction occurs in the outside of it. In the inside of the glass, the eraser is considerably distorted and the refractive image can be displayed in real time according to the eraser movement. In the simulation, we used a normal PC that has Intel Corei7 3.4GHz CPU, 8GB main memories, NVIDIA GeForce GTX570 GPU, and measured the performance of the method. For the final image with 1,600 × 1,200 resolution, the average of the frame rate by the proposed method was 96 FPS (Frames Per Second) to display refracted objects that are composed of 100,000 vertices, while that by GPU based ray tracing method was 26 FPS.



(a) CG simulation (b) Real refraction

Figure 1. Comparison of the refractive image by CG simulation and that by real phenomenon.

Keywords: *Computer graphics, physical simulation, ray tracing, refraction*

1. INTRODUCTION

Computer graphics is usually used to generate realistic images based on physical phenomena, and one of the most challenging issues is a physical simulation based on optical phenomena such as reflection and refraction. The most simple and powerful method is ray tracing that generates realistic images by tracing the ray input into the eye (Whitted, 1980). Ray tracing method, however, takes huge time to generate an image so that texture mapping method had been developed by (Blinn and Newell, 1976) before ray tracing was invented. For reflection, texture mapping sometimes does not work well especially for curved objects. Then, (Ofek and Rappoport, 1998) proposed an interactive method of reflection for curved objects by generating virtual objects, which are previously transformed according to the reflection. On the other hand, (Hakura et al., 2001) proposed parameterized environment map that can reproduce local reflections from pre-rendered viewport samples. (Iwasaki et al., 2003) proposed a fast rendering method for refractive and reflective caustics due to water surfaces by generating texture mapped slices and by using reflection and refraction mappings of them. (Wang et al., 2004) also proposed real-time interpolation method of environment map by using depth buffer. In addition, (Yamashita et al., 2003) proposed another method for measurement of objects in liquid by calculating 3D coordinates of object surfaces with a camera and a spot laser beam. Nowadays, GPU (Graphics Processing Unit) is faster than CPU (Central Processing Unit), and GPU can perform many calculations that CPU usually operates as well as graphics processing such as geometry transformation and object rendering. With the background, (Wyman, 2005) proposed an interactive image-space approach for refraction by utilizing GPU. (Hu and Qin, 2007) also proposed an interactive computing scheme where the calculation of ray-object intersection is computed on GPU for rendering of reflection, refraction and caustics. In addition, (Oliveira and Brauwers, 2007) proposed another image-space technique that calculates ray-object intersection efficiently by using programmable GPU. Furthermore, (Rodgman and Chen, 2006) proposed another method to render refraction in volume graphics, and (Rousiers et al., 2012) proposed a method to render refraction through a transparent material with rough surfaces that cause wide scattering. On the contrary, (Chen and Wong, 2013) proposed a depth acquisition method from refracted images. In this method, the depth from the eye can be calculated, if the transparent material that causes refraction is known. This means that we can geometrically calculate the shape of the object deformed by refraction if the transparent material is previously known.

Therefore, we propose a new method for rendering refraction, which geometrically calculates the movement of vertices constructing the objects deformed by refraction for a cylindrical glass (Makino et al., 2012). Refraction time depends on the shape of the transparent material and the ray path. Here, as (Ofek and Rappoport, 1998) proposed, it is very efficient to generate virtual objects for interactive reflection rendering. Then, in our method, virtual objects are generated by geometrical calculation that depends on the material shape and refraction time, and the final images are generated by combining some images generated with virtual objects. The refraction calculation is based on Snell's law so that the result is accurate. In addition, the proposed method is very fast and the image can be generated in real time because the calculation is performed only for vertices constructing the display objects instead of pixels constructing the display image. This paper describes the detail of the method and shows some applications for a parallel glass and a cylindrical glass.

2. METHOD

2.1. Calculation for Parallel Glass

Figure 2 shows the diagram of refraction through a parallel glass. The figure has a coordinate system, where the eye is the origin and X and Y axes are directed horizontally to left and vertically down, respectively. In the figure, n and n' are the refractive indices of glass and air, respectively, and W is the thickness of the glass. The ray that outputs from the eye injects into the glass with the incidence angle θ, and is refracted at the incidence point, where ϕ is the refraction angle. The refracted ray outputs at a point of the bottom of the glass, which is indexed as A' in the figure, when it is refracted again with the emergence angle that is the same as the incidence angle. The ray reaches a point of A(x, y), and A is seen from the eye by refraction. Then, if A

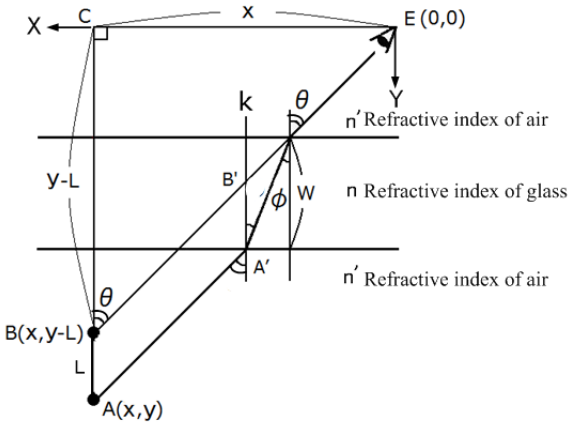


Figure 2. Refraction through a parallel glass.

moves to B, which is an intersection point of the ray that is not refracted and the vertical line that passes A and is parallel to Y axis, we can see the refracted image without the parallel glass. Suppose that the length between A and B is L and A is written as A(x, y). Then, B can be written as B(x, y-L). Here, B' is defined as the intersection point of the ray that is not refracted and a line k that passes A' and is parallel to Y axis. Then, the length between A' and B' becomes L. Figure 3 shows the expanded diagram of Figure 2. From Figure 3, the following equation is derived.

$$W - L = W(\tan\phi)\tan\left(\frac{\pi}{2} - \theta\right) \quad (1)$$

Then, L can be calculated with the following equation.

$$L = W\left\{1 - (\tan\phi)\tan\left(\frac{\pi}{2} - \theta\right)\right\} \quad (2)$$

However, θ is the incidence angle for the point A that constructs a display object and it depends on the positions of the eye. On the other hand, the following equations are true from △ ECB in Figure 2 and Snell's law.

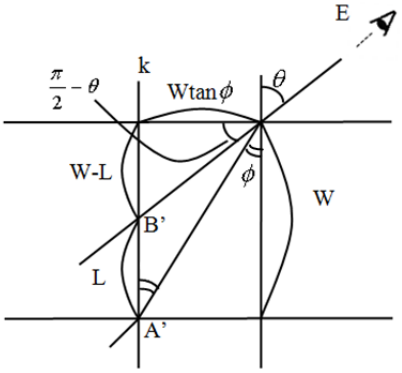


Figure 3. Expanded image of parallel refraction.

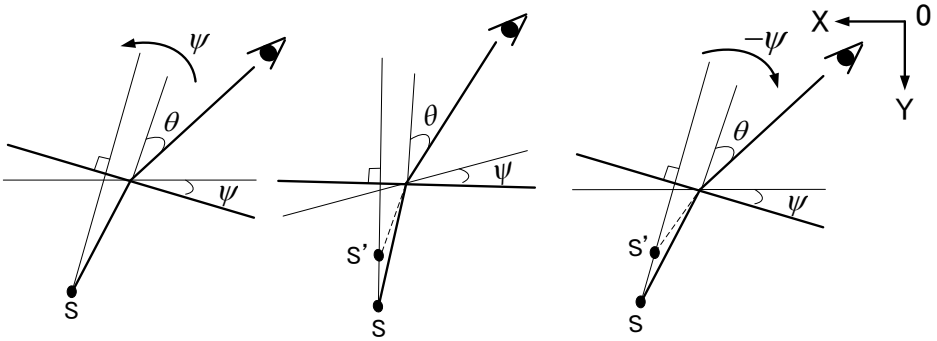
$$\tan\theta = \frac{x}{y - L} \quad (3) \qquad \frac{n}{n'} = \frac{\sin\theta}{\sin\phi} \quad (4)$$

By substituting Equation (3) and Equation (4) to Equation (2), the following equation is derived.

$$(y - W)\tan\theta + W\tan\left\{\sin^{-1}\left(\frac{n'}{n}\sin\theta\right)\right\} = x \quad (5)$$

Finally, by solving Equation (5) with an iterative method, the incidence angle θ is calculated. Once θ is calculated, ϕ is also calculated with Equation (4). Then, the refracted object can be generated by moving A(x, y) to B(x, y-L) with Equation (2).

2.2. Calculation for Inclination Angle



(a) Rotation (b) Movement (c) Inverse rotation

Figure 4. System rotation before the movement calculation.

Figure 2 and Figure 3 show the diagrams of the case that the surface of the parallel glass is parallel to X axis. However, the surface of the glass is usually inclined against X axis. The equations that have been derived in 2.1 are true for the case that the surface of the glass is parallel to X axis. Otherwise, the rotation of the system is required before the above equations are applied. Figure 4 shows the diagram for the case that the surface of the glass is not parallel to X axis. In Figure 4(a), the surface is inclined against X axis by the angle ψ. By the rotation, the surface becomes parallel to X axis and the movement of S can be calculated by applying the equations derived in 2.1. Then, the final position S' is solved with the inverse rotation by the angle $-\psi$.

2.3. Calculation for Cylindrical Glass

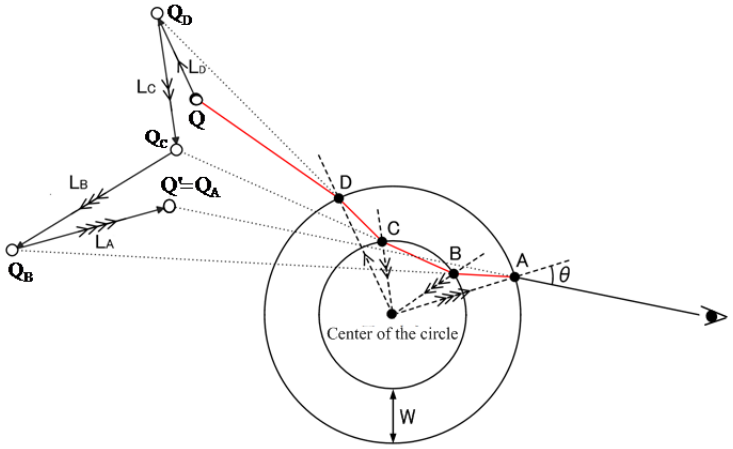


Figure 5. Refraction through a cylindrical glass.

Figure 5 shows the diagram for the refraction through a cylindrical glass that has the thickness of W. In the refraction through a cylindrical glass, four-time refractions occur at the most for the scene that is seen in the inside of the glass. In Figure 5, the point Q is seen by four-time refractions. Therefore, if Q can be moved to Q', we can see the image refracted through the cylindrical glass without refraction calculation. For the movement, the ray that outputs from the eye passes four points in the sequence of A, B, C and D, and the point constructing the display object moves from Q to Q' in the inverse sequence of Q_D, Q_C, Q_B and $Q'(=Q_A)$. This refraction can be easily explained by moving the eye in the sequence of C, B, A, and the original eye position. Suppose that the eye is located at the point of C. Then, the ray output from C is refracted at D. In this case, the refraction occurs once.

Figure 6 shows another refraction diagram similar to Figure 3 but refraction occurs only once, while refraction occurs twice in Figure 3. In Figure 6, n_A and n_B are the refractive indices of material A and B, respectively. For example, the material A is the glass and the material B is the air in the case that the eye is located at C in Figure 5, while the material A is the air and the material B is the glass in another case that the eye is located at B in Figure 5. In Figure 6, the ray that outputs from the eye is refracted with the incidence angle of θ and the refractive angle ϕ. P(0, L) is the point that can be seen through the refraction and P'(0, L') is the point that is seen without refraction. By moving P to P', the refracted image can be generated without refractive calculation. From Figure 6, the following equation is true.

$$L\tan\phi = M = L'\tan\theta \qquad (6)$$

The following equation is also true by Snell's law.

$$\frac{n_B}{n_A} = \frac{\sin\theta}{\sin\phi} \qquad (7)$$

From Equation (6) and Equation (7), the following equation is obtained.

$$L = \frac{n_B \cos\phi}{n_A \cos\theta} L' \qquad (8)$$

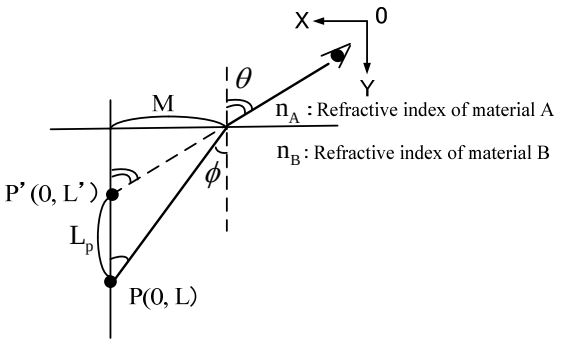


Figure 6. One time refraction between two materials.

Then, the movement length of $L_p = L - L'$ can be calculated, where θ and ϕ vary depending on the positions of the incidence point and the eye so that θ should be calculated with the iterative method for Equation (5), and ϕ is calculated by using Equation (7).

Here, let us go back to the explanation of Figure 5. If the eye is located at C, the ray is refracted at D and the point constructing the display object moves by the length of L_D, where the movement direction is parallel to

the line that passes the center of the circle and D because the movement direction in Figure 6 is parallel to Y axis. In addition, in Figure 6, P' is the intersection point of the line that passes P and is parallel to Y axis, and the ray output from the eye without refraction. Therefore, Q_D in Figure 5 is the intersection point of the line that passes Q and is parallel to the line that passes the center of the circle and D, and the ray output from C without refraction. Actually, A(x, y) in Equation (5) is the relative coordinate to the eye potion, which is C in this case. Then, the eye position is temporary decided and solved with the iterative method for the line that passes Q_D and D to pass the eye point C. In the same manner, Q_D moves to Q_C as the eye moves from C to B, and Q_C moves to Q_B as the eye moves from B to A. Finally, Q_B moves to $Q'(= Q_A)$ as the eye moves from A to the original eye position. As the result, we can generate the refracted data by moving Q to Q' and obtain the refracted image by rendering some images generated according to the refraction times.

2.4. Division of the Display Area

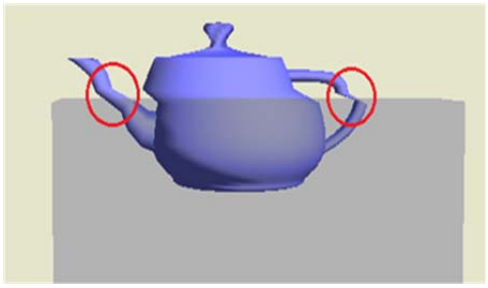


Figure 7. An image generated by vertex interpolation.

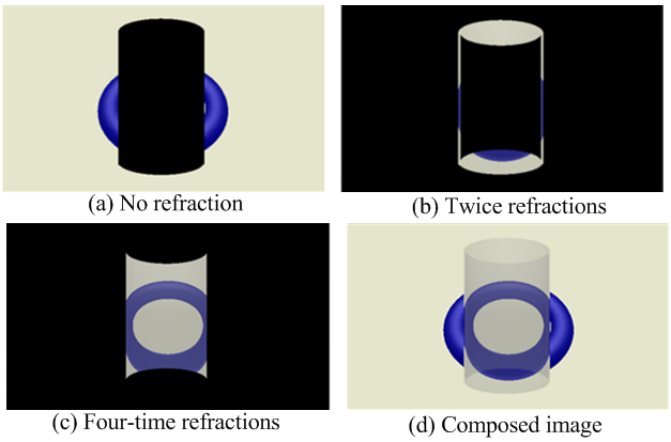


Figure 8. Division of the area and the composed image.

This method performs refractive calculation not for the pixels that construct the display image, but for the vertices that construct display objects. Therefore, the intermediate point between vertices is interpolated with the result calculated for vertices. Figure 7 shows the image that is generated by interpolation between vertices that have been calculated for refraction. The images in the areas circled with red are distorted. In order to avoid this defect, some refractive images should be generated according to the refraction times, and they should be composed pixel by pixel. For image composition, the display area should be divided according to refraction times. In order to investigate the refraction times, depth peeling method (Nakagawa et al., 2011) is used, which peels the surfaces that construct display objects. The method can investigate the number of peeled surfaces per pixel constructing the display image, and the number corresponds to the refraction times for the pixel. Thus, we can generate the precise refractive image by composing some images pixel by pixel, which are generated according to the refractive times. Figure 8 shows some areas that have been divided according to the refractive times. Only the black area is excluded for rendering. For example, Figure 8(a) shows that only the area of the cylindrical glass is excluded for no refraction. In other words, black area has several times of refractions. For refraction through a cylindrical glass, there are three kinds of refractions: once, twice, and four times. Figure 8(d) shows the final image generated by composing three kinds of refractive images: Figure 8(a), (b) and (c).

3. SIMULATION

3.1. Simulation Result for a Parallel Glass

Figure 9 shows the results of the simulation with parallel glasses. Figure 9(a) and (b) show the results of refraction through circle shaped parallel glasses that have the inclination angle of 5° and 30°, respectively. From the figures, we can understand that the spout of the teapot is refracted. In fact, the gap depends on the inclination angle of the glass, and it in Figure 9(b) is larger than that in

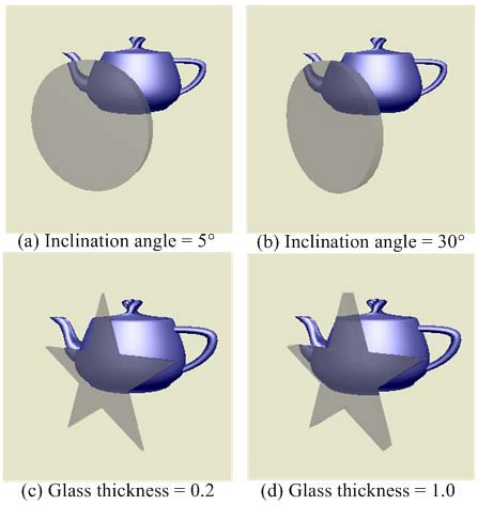


Figure 9. Results of the simulation with parallel glasses.

Figure 9(a). On the other hand, Figure 9(c) and (d) show other results of refraction with star shaped parallel glasses that have the thickness of 0.2 and 1.0, respectively. From the figures, we can also see that the gap depends on the thickness of the glass, and it in Figure 9(d) is larger than that in Figure 9(c). In addition, we have confirmed that the image generated by this method is very similar to that by ray tracing. For the simulation, we used a PC that has Intel Core i7 3.4GHz CPU, 8 GB main memories, and NVIDIA GeForce GTX570 GPU. This method performs refractive calculation only for vertices constructing the display objects instead for pixels constructing the display image. Therefore, we have investigated the relation between vertex number and FPS (Frames Per Second). Table 1 shows the evaluation result. From the table, real-time simulation, which is more than 30 FPS, is possible if display objects are composed with less than 600,000 vertices.

Table 1. Relation between vertex number and FPS (simulation with a parallel glass)

Vertex number	FPS
100,000	122
200,000	81
300,000	55
400,000	42
500,000	36
600,000	30
700,000	26
800,000	22

3.2. Simulation Result for a Cylindrical Glass

Figure 1(a) shows the result of the refractive simulation with a cylindrical glass cup, while Figure 1(b) shows the real refraction. By comparison of the figures, we can understand that the eraser in the inside of the glass is refracted, while the eraser in the outside of the glass is not refracted. In the simulation, two refractions occur at the verge of the cylindrical glass, where both of refraction and reflection occur, and the reflection has great influence on the image as Figure 1(b) shows. Then, this time, the refraction in the inside of the verge was not considered in the simulation. Table 2 shows the performance evaluation result that compares the FPS by this proposed method with that by OptiX, which is ray tracing based rendering method that uses GPU. The same PC was used for the simulation, and the resolution of the display image was 1,600×1,200. The FPS by the proposed method is not constant for an object that has a constant number of vertices because the iteration time varies depending on the incidence angle. Therefore, Table 2 shows the worst and the best cases, and also the average of FPS. From Table 2, the average FPS was 96 for 100,000 vertices, while that was 122 for the refraction through a parallel glass. This is because that the refraction time with a cylindrical glass is larger than that with a parallel one so that many iteration times are required to solve the equations. The proposed method performs the refractive calculation for vertices constructing the display objects so that FPS is gradually lower as the number of vertex increases, while OptiX performs the calculation for pixels constructing the display image so that the FPS is almost constant.

On the other hand, Table 3 shows another result of the relation between pixel number and FPS. In the simulation, a model constructed with 100,000 vertices was used. OptiX performs the refraction calculation per pixel that constructs the display image so that the FPS decreases as the resolution is improved, while the FPS by the proposed method is almost constant.

Table 2. Relation between vertex number and FPS (simulation with a cylindrical glass) [FPS]

Vertex number	Proposed method			OptiX
	Worst	Best	Average	
100,000	65	130	96	26
200,000	37	75	56	25
300,000	24	54	39	24
400,000	18	42	30	22
500,000	15	34	25	21
600,000	12	28	20	21
700,000	10	24	17	20
800,000	9	22	16	20

Table 3. Relation between pixel number and FPS (simulation with a cylindrical glass) [FPS]

Resolution	Proposed method	OptiX
640×480	98	60
800×600	98	45
1,024×768	97	34
1,600×1,200	96	26

4. DISCUSSION AND CONCLUSIONS

This paper has proposed a new refraction method that previously generates some kinds of refracted images by calculation for the position of vertices constructing display objects, and composes them to create the final image refracted through a parallel or cylindrical glass. In order to calculate the movement vector of each vertex, an equation is derived based on Snell's law. This equation can calculate the each vertex position

refracted by a transparent matter; however, the incidence angle needs to be solved by an iterative algorithm. In addition, the surface of the transparent material is not always parallel to an axis of the coordinates. In this case, the system should be rotated for the surface to be parallel to an axis of the coordinates, and it should be rotated back with the inverse rotation after the refractive calculation. On the other hand, refractive calculation through a cylindrical glass is almost the same method; however, there are four-time refractions in the inside of the cylinder. For the refraction, the movement vector of a vertex needs to be solved with four-time calculations. In addition, the proposed method performs only for vertices constructing display objects so that the pixels between vertices need to be interpolated. If interpolated pixels between a refracted vertex and a vertex with no refraction are displayed, the image is distorted. Then, some images should be generated according to refraction times, and they need to be composed according to the area that depends on refraction times, which can be investigated by depth peeling method.

As the results of the simulations, we have confirmed that images are refracted by being seen through a parallel glass, and the gap between a refracted part and another with no refraction depends on the inclination angle and the thickness of the parallel glass. For the simulation with a cylindrical glass, we have confirmed that the refraction generated by the simulation is accurate by comparing a generated CG image with a real refraction. For the performance evaluation, the proposed method is so fast that real-time simulation is possible. Especially, the FPS of this method is almost constant even if the resolution of the display image increases, while the FPS of a ray tracing based method with GPU decreases as the resolution increases.

The refracted image seen through a cylindrical glass has been generated in real time. However, there are two refractions at the verge of the cylindrical glass, and the reflection has a great influence of the final image so that the refraction at the verge of the glass was not included in the simulation. Then, in the future, we have to include reflection as well as refraction in the method. In addition, the proposed method can calculate the movement vector of vertex if the shape of the transparent matter is geometrically defined. Then, as the issue of the future works, we have to come up with a solution for arbitrary shaped object that can be defined as spline or Bézier surface.

REFERENCES

Blinn, J. F. and Newell M. E. (1976). Texture and reflection in computer generated images. *Communication of the ACM*, 19(10), 542–547.

Chen, Z. and K. K. Wong (2013). Depth from refraction using a transparent medium with unknown pose and refractive index. *International Journal of Computer Vision*, 102(1-3), 3-17.

Hakura, Z. S., J. M. Snyder, and J. E. Lengyel (2001). Parameterized environment maps. *Proceedings of the 2001 symposium on Interactive 3D graphics*, 203–208.

Hu, W. and K. Qin (2007). Interactive approximate rendering of reflections, refractions, and caustics. *IEEE Trans. on Visualization and Computer Graphics*, 13(1), 46–57.

Iwasaki, K., Y. Dobashi, and T. Nishita (2003). A Fast Rendering Method for Refractive and Reflective Caustics Due to Water Surfaces, *Computer Graphics Forum*, 22(3), 601-609.

Makino, Y., Y. Chang, and N. Mukai (2012). Fast refraction method based on geometrical calculation through a cylindrical glass. *Proceedings of NICOGRPH 2012*, 91-94.

Nakagawa, M., N. Mukai, and M. Kosugi (2011), A fast collision detection method by using modified depth peeling, *The journal of the Institute of Image Information and Television Engineers*, 65(6), 825-832.

Ofek, E. and Rappoport, A. (1998). Interactive reflections on curved objects. *Proceedings of the 1998 SIGGRAPH*, 333-342.

Oliveira, M. M. and M. Brauwers (2007). Real-time refraction through deformable objects. *Proceedings of the 2007 symposium on Interactive 3D graphics*, 89–96.

Rodgman, D. and M. Chen (2006). Refraction in volume graphics. *Graphical Models*, 68, 432-450.

Rousiers, C., A. Bousseau, K. Subr, N. Holzchuch, and R. Ramamoorthi (2012). Real-time rendering of rough refraction. *IEEE Trans. on Visualization and Computer Graphics*, 18(10), 1591–1602.

Wang, W., L. Wang, S. Lin, J. Wang, and B. Guo (2004). Real-time environment map interpolation. *Proceedings of the 3rd International Conference on Image and Graphics*, 382–389.

Whitted, T. (1980). An improved illumination model for shaded display. *Communication of the ACM*, 23(6), 343–349.

Wyman C. (2005). Interactive image-space refraction of nearby geometry. *Proceedings of the 3rd international conference on Computer Graphics and Interactive Technique in Australasia and South Asia*, 205–211.

Yamashita, A., E. Hayashimoto, T. Kaneko, and Y. Kawata (2003). 3-D Measurement of Objects in a Cylindrical Glass Water Tank with a Laser Range Finder, *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1578-1583.