Received February 26, 2013; Accepted June 24, 2013

International Journal of Asia Digital Art&Design

Generating Wayang Beber of Pacitan Character's Outline Using Renderman Interface

Grahita, Banung
Tokyo Metropolitan University
banung-grahita@sd.tmu.ac.jp

Komma, Toshihiro
Tokyo Metropolitan University
komma@sd.tmu.ac.jp

Kushiyama, Kumiko
Tokyo Metropolitan University
kushi@ea.mbn.or.jp

Abstract
Wayang is a traditional medium for storytelling in Indonesia. Wayang isn't popular today. The Digital technology gives opportunities to develop new types of wayang which could be interesting to the new generation. Our research attempts to develop a new form of wayang using 3D computer graphics animation technology. We want to make a CG animated wayang based on the visual form of wayang beber especially wayang beber of Pacitan.
Wayang beber is a distinct type of wayang. Unlike the other types of wayang, wayang beber is not a puppet, but a sequence of picture drawn on several scrolls. Wayang beber is known as the oldest type of wayang. One of the wayang beber types is wayang beber of Pacitan.
Wayang beber of Pacitan especially the character figure has a unique visual form. It is a flat 2 dimensional graphic with limited colors but rich with ornamental details. We use a non-photorealistic rendering technique to simulate the wayang beber of Pacitan visual features on a 3D CG model. One of the important wayang beber of Pacitan visual features is the outline.The wayang beber of Pacitan outline has some incorrectness qualities, such as a variation of thickness and wiggliness. These qualities are typical characteristics of a human drawing. The first step of our research is generating an outline of a 3D CG figure that has these incorrect qualities. In this paper we explain the development of the shader for generating the desired outline qualities that could work in the Renderman interface. We developed a shader algorithm that can displace an object's surface point with the gradual and random distance. This algorithm can be used to generate an outline with a variation of thickness and wiggliness.

Keywords: computer graphic, animation, non-photorealistic rendering, renderman, traditional picture

1. Introduction

Wayang is a traditional medium of storytelling from Indonesia. According to Hazeu [1], wayang originated from the pre-historic era of Java in Indonesia. Wayang evolved for many centuries as a tool for religious ceremonies, affected by Hindu and Islamic culture.

Wayang was very popular in Indonesia, particularly in Java. This was realized by the wali, Islamic leader of Java in the 16th century. They used wayang to disseminate Islamic religion in Java [2]. The popularity of wayang also can be seen in the late 18th century; in this era many Javanese traditional illustrations were influenced by the wayang visual style [3].

The popularity of wayang is decreasing today and several types of wayang are rarely performed. Some of them even can no longer be seen. The younger generation seems uninterested in the old form of wayang. Digital technology gives opportunities to develop new types of wayang which is more interesting to this younger generation.

Our research attempts to develop a new version of wayang using 3D computer graphics animation technology. We want to make a CG animated wayang based on the visual form of wayang beber especially wayang beber of Pacitan.

Wayang beber is one type of wayang. It is a sequence of pictures drawn on several scrolls. One type of wayang beber that remains today is wayang beber of Pacitan[4]. Wayang beber of Pacitan tells the story of Jaka Kembang Kuning, so it is also known as wayang beber Jaka Kembang Kuning (see Fig. 1).

We choose to develop wayang beber because it is the origins of animation. As we know, the sequential drawing of human or animal figures such as wayang beber has been considered as the origins of animation [5]. Wayang beber of Pacitan especially the characters also has a unique visual form. It is a flat 2-dimensional graphic with limited colors but rich with ornamental details. Compare to modern pictures, wayang beber of Pacitan has a distinct way of depicting human figure.

Figure1. Part of a fifth wayang beber of Pacitan scroll

We use 3D CG technology because of several advantages, such as free inbetweening, free perspective, unlimited revisions, and an economy of scale [6]. Nowadays, the 3-dimensional computer graphic animation technology is commonly used to produce 2-dimensional graphics using a process called non-photorealistic rendering. This process needs several steps. This paper is only focusing on the first step: render the outline of the 3D computer graphic figure. We rendered the image using Renderman, an Application Programing Interface designed by Pixar studios for animation production. In this paper we explain the development of the shader for generating the wayang beber of Pacitan outline on the 3D CG figures that can be worked in the Renderman interface.

2. The Features of Wayang Beber of Pacitan Outline

According to Curtis, the most important thing in non-photorealistic rendering is defining the art direction. The art direction is the mental image of what the finished product should look like, down to the tiniest detail [6]. To define the art direction, we needed a good reference image. We used a replica of the wayang beber of Pacitan painting as the reference image. This replica is made by a modern artist, a member of the wayang beber metropolitan community in Jakarta, Indonesia (see Fig.2).

We used the replica because the original wayang beber of Pacitan painting is difficult to access. The original painting is believed as a sacred thing by the local community. It can only be seen in a certain occasion. We have access to a representative copy of wayang beber of Pacitan painting, as can be seen in figure 1. This copy is made by royal painter by the order of the King of Mangkunegaran in 1939. This painting is kept in Mangkunegaran Palace, Surakarta city, Indonesia. However the condition of the painting is not good, so we needed to look for another painting as a reference image.

We found a replica of the wayang beber of Pacitan painting made by a member of wayang beber metropolitan community. The quality of the painting is very good and imitates the original painting very well. Therefore we used this painting as a reference image.

Figure2. The replica of the wayang beber of Pacitan painting [7].

We examined the reference image using a theory by Professor Primadi Tabrani from Bandung Institut of Technology (ITB), Indonesia [4]. Professor Primadi Tabrani studied several Indonesian traditional pictures including the wayang beber of Pacitan and then developed a theory called visual language. This theory described the traditional picture as visual language; the picture doesn't only represent an object but is also used to communicate a message or story. The most important concepts in visual language theory are the image and the visual grammar. The image is the smallest unit in visual language; it is equal with words in verbal language. The visual grammar is certain rules to form, organize, or arrange the image in order to be able to communicate the message.

Since this research focused on the wayang beber of Pacitan character figure, we only needed to understand the image. In visual language theory the image consists of the image content and the image way. The image content is the object which is depicted by the image, and the image way is the way to draw the image. For example, we took a look at the reference image of Gandarepa (see Fig.3). The image content of this image is the character of Gandarepa, one of the protagonist characters in the wayang beber of Pacitan story. The image way of this image is the way to draw the character of Gandarepa.

Professor Primadi Tabrani discovered many ways to draw the image in traditional pictures. He classified the ways into four types: the drawing size, the drawing angle, scale and the drawing technique. In this paper we focus on the drawing technique.

Figure3. the image of Gandarepa character

The drawing technique is a way to draw the image using the visual elements, such as line, color, and shape. According to Primadi Tabrani, the Gandarepa image in wayang beber Pacitan is drawn using a drawing technique called "blabar" [4]. Blabar is the way to draw the image by using lines to form the shape. This drawing technique uses limited colors.

In another literature, we found that wayang beber of Pacitan is made using Javanese traditional painting technique called "sungging" [8]. The complete explanations of sungging painting technique can be found in the paper by Ahmadi [9], Purbasari [10], and the book by Sukir, written in 1980[11].

Basically, this sungging technique has similarities with the blabar technique which is explained by Primadi Tabrani. Purbasari[10] in her paper said that the sungging technique begins by coloring the figure, and is followed by adding the outline on the image. The outline is used to form and fix the image shape.

The sungging technique has certain rules to make the painting. These rules cause distinct qualities of lines, colors, and patterns; we described them as visual features of wayang beber of Pacitan. We need to simulate all of the visual features in computer graphics to create a wayang beber of Pacitan animated film. However, in this paper we focus on the outline features. The complete explanation about the other features can be seen in [12].

The outline of the wayang beber character image has three important features. Firstly, the outline is relatively thick. There is no literature explaining how thick the outline should be made. However the outline should be thick enough so it can be seen clearly from a distance. The thick outline has a function to emphasize the character shape, and to separate it from the background.

Secondly, the outline has a variation of thicknesses (see Fig.4.). The outline thickness, usually called the line width, is not uniform. The artist paints the outline by hand. During the painting process the pressure by the artist's hand on the brush may change, and as a result the line width is changed (see Fig.5a). Thomas Strohotte and Stefan Schlechtweg described this as the incorrectness qualities of human drawing [13].

The third feature is wiggliness. Wiggliness is some irregularities that occurred on the outline shape (see Fig.4.). This is also one of the incorrectness qualities of human drawing. The lines drawn by a human artist are never completely straight, especially if the artist doesn't use any aids, such as a ruler. The lines are more or less wiggly (see Fig.5b). The wiggliness is caused by a small irregular movement of the artist's hand when he or she is drawing the image; the wiggliness also can be caused by the paper structure [13].

We intended to generate the wayang beber character using 3D CG. In contrast with a human drawing, 3D CG generate perfectly smooth and straight lines with a uniform thickness, as we see in the works of Hajagos [14], De Wolf [15], and Apocada [16]. Therefore we need a new algorithm to generate a line with incorrectness qualities as we see in wayang beber of Pacitan.

Figure4. Outline of Gandarepa image

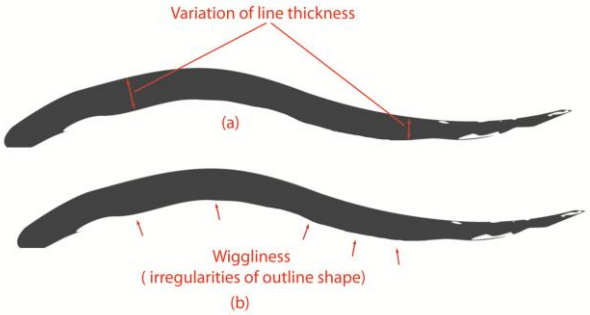


Figure5. Sample of hand drawn line. (a) Variation of thickness in hand drawn line. (b) Wiggliness in hand drawn

3. Related Works

This research tried to develop a shader that could generate the wayang beber of Pacitan outline in a 3D CG model and worked on Renderman interface. The reasons why we use the Renderman are: it can produce a high quality image, it has great programmability (great amount of control) and it is designed for animation.

There are two shaders in the Renderman interface that can be used to generate an outline. The first is cel shaders, developed by Apocada and Gritz [16], and the second one is ID outline shader, developed by Ivan De Wolf [15]. However these two shaders don't fit our purpose. Although they can generate a variation of thickness, the Apocada and Gritz's shader sometimes produce an undesired dark area on the rendered image (see Fig.6a). Meanwhile, Ivan De Wolf's ID outline shader can produce a good and clean image, but it doesn't generate any variation of thickness and wiggliness (see Fig.6b). Therefore we need a new shader to generate the outline.

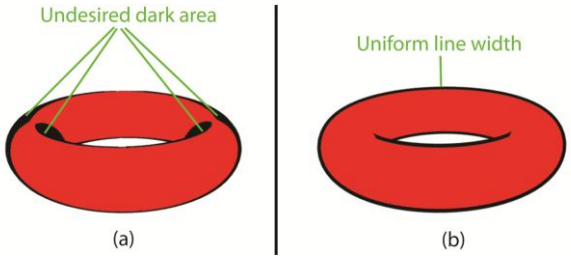


Figure6. (a) Rendering result of Apocada and Gritz's shader. (b) Rendering result of Ivan De Wolf's shader

The important concept for creating an outline in a 3D CG object is the silhouette edges. The outline is created by finding and displaying the silhouette edge. One of the methods to do this is two-pass rendering. This method was used by Gooch [17] and Raskar [18].

The two-pass rendering method works in the image space. This method uses a two layer set of polygons; the first layer is a layer of front-facing polygons and the second layer behind is a layer of backfacing-polygons. Back facing-polygons are rendered first, and then the front-facing polygons are rendered on top. The intersection of these two layers in the image space creates silhouette edges. Raskar and Cohen increased the area of the intersection by pulling the back-facing polygons slightly forward towards the camera [18]. This method is easy to

implement. The time to write and debug the code was very short, the visual quality was good, the speed was fast, and the method scaled well.

We are interested in Raskar and Cohen's method to render silhouette and are looking forward to doing a similar approach that can be applied in the Renderman interface. As we found, Ivan De Wolf's ID outline shader has quite a similar algorithm to Raskar and Cohen's. However, instead of using two layers of images in the image space, De Wolf's algorithm used two objects in the object space. Ivan De Wolf's algorithm used two identical objects; one of the objects was scaled and rendered as the outline. This shader can produce a thick outline on the 3D graphic figure. However it still can't resemble the incorrectness qualities of a human drawing. We create a new shader based on the algorithm of the ID outline shader. Our new algorithm was implemented in Renderman Shader Language (RSL) to develop a new outline shader for Renderman.

4. Generating The Wayang Beber of Pacitan Character's Outline

To generate the outline, firstly, we duplicated the object, and then we labeled them as object A and B. A became the outline and B became the fill. Our approach had some similarities with Rascar and Cohen's two-pass method [18]. However, instead of rendering two layers of images, our approach rendered two identical object in object space. Next we will explain the process, beginning with the fill.

The fill consists of one solid color, and has a wiggly shape. We rendered the fill (object B) using a shader with the following algorithm.

Creating the random wiggliness

1. Input s and t; the texture coordinate.
2. Input W; the frequency of wiggliness
3. Scale the s and t by multiplying them with W.
4. Apply Perlin noise function to scaled result of S and T to get a random number (H).
5. Input K; the amplitude of the wiggliness
6. Scale H by multiplying it by K
7. Calculate Nn by normalizing the surface normal.
8. Randomly displace the surface point P in the direction of its normal

$$\vec{P}' = \vec{P} - \vec{N}n \times H \times K$$

Shade the fill color

9. Input Cs; the fill color
10. Input Os; the fill opacity
11. compute the Ci output color
    Ci = Cs × Os

Then we rendered the outline (object A) using a shader with the following algorithm.

Displace the surface

1. Input width
2. Calculate Nn by normalizing the surface normal.
3. Add the new vector V
4. **Compute D; the dot product between normalized V and Nn.**
5. **Input MinWidth; the minimum width value of the line**
6. Compute S; the parameter to create a displacement ramp

   S= (1-D) + D × MinWidth
7. Displace the surface point P in the direction of its normal

$$\vec{P}' = \vec{P} + \vec{N}n \times (\text{width} + (\text{width} \times S))$$

Creating the random wiggliness

8. Input s and t; the texture coordinate.
9. Input W; the frequency of wiggliness
10. Scale the s and t by multiplying them with W.
11. Apply Perlin noise function to scaled result of S and T to get a random number (H).
12. Input K; amplitude of the wiggliness
13. Scale H by multiplying it by K
14. Randomly displace the surface point P in the direction of its normal

$$\vec{P}' = \vec{P} - \vec{N}n \times H \times K$$

Shade the outline

15. Detecting silhouette edges by computing D1, the dot product between Nn and normalized I; the viewing direction vector.
16. If D1 > 0, then set the output opacity to 0
    If D1 < 0, then set the output opacity to 1
17. Input Cs; the outline color
18. Set the Output color
    Ci = Cs;

We wrote our algorithm based on the ID outline algorithm [14]. To explain our algorithm first we will explain about the ID outline algorithm.

4.1. ID Outline Algorithm

In the ID outline algorithm, the surface of A is displaced by moving the surface point in the direction that is parallel to its normal. Normal is a vector that describes the surface orientation. The surface normal can have varying length, so it must be normalized to ensure its length doesn't affect the displacement.

In the displacement process, the surface point is moved with a distance that is equal to the value that is defined as line width. The process described in the following equation:

$$\vec{P}' = \vec{P} + \vec{N}n \times \text{width} \qquad (1)$$

Where $\vec{P}'$ is the new surface point, $\vec{P}$ is the old surface point, $\vec{N}n$ is the normalized normal, and the width is the displacement distance.

After displacement, the displaced object (object A) will be bigger than the other object (object B). Since the position coordinate of the two objects are exactly the same, object B will be placed inside object A, so object B will be invisible (see Fig 7).

To make object B visible, the front facing polygons of object A must be invisible.

The orientation of the object A polygon can be detected by computing the dot product between its surface normal and the viewing direction vector [19]. This process is described in

$$\vec{N}n \cdot \vec{I}n = |\text{Nn}|\,|\text{In}|\cos(\theta) \qquad (2)$$

Where $\vec{N}n$ is the normalized normal, $\vec{I}n$ is the normalized viewing direction vector, and θ is the angle between two

vectors.

If the dot product is positive then the polygon is back facing, otherwise if the dot product is negative then the polygon is front facing.

After the polygon orientation is known then the opacity of the polygon can be set. The back-facing polygon opacity was set as 1, so it was visible, and the front-facing polygon opacity was set as 0, so it was invisible.

As a result the object B that is placed inside object A was visible. The viewer will see the object A's back-facing polygon as the object's outline (see Fig. 8).

.

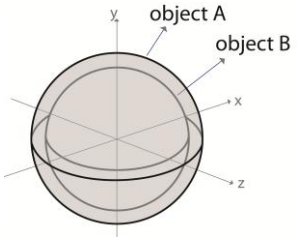


Figure7. Object B was placed inside object A after displacement

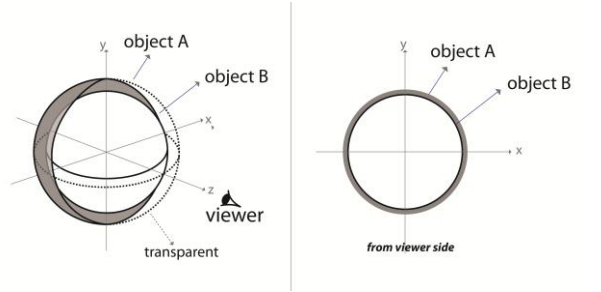


Figure8. The front facing Polygon of object A is invisible; object B inside object A is visible

4.2. Creating Variations of Line Thickness

The ID outline shader generate a smooth outline with a constant thickness, but the outline in the Wayang Beber of Pacitan character has a variation of thickness and wiggliness. We modified the IDoutline shader to produce these qualities. To produce a variation of thickness, every surface point in object A should not move in the same distance. We created a displacement ramp based on the surface point position. Using a displacement ramp the distance of surface point displacement could be changed gradually.

To create a displacement ramp, we added a new vector as the parameter. The direction of the new vector can be various. We can define the x and y coordinate, but the z coordinate must be 0. In this paper we added a vector that pointed downward, perpendicular with the x axis, as seen in Fig.9.

Then we computed the dot product between the normalized new vector ($\vec{V}n$) and the normalized surface normal ($\vec{N}n$) as in

$$D = \vec{V}n \bullet \vec{N}n \qquad (3)$$

Where D is the dot product between the normalized new vector ($\vec{V}n$) and the normalized surface normal ($\vec{N}n$), $\vec{V}n$ is the normalized new vector, and $\vec{N}n$ is the normalized normal.

If the normalized new vector ($\vec{V}n$) and the normalized surface normal ($\vec{N}n$) are perpendicular, the dot product will be 0 and if they are parallel the dot product will be 1.

The new parameter to create the displacement ramp was obtained using the following equation:

$$S = (1-D) + D \times 0.1 \qquad (4)$$

Where S is the new parameter to create the displacement ramp.

Then the displacement ramp was created using the following equation:

$$\vec{P}' = \vec{P} + \vec{N}n \times (\text{width} + (\text{width} \times S)) \qquad (5)$$

Where $\vec{P}'$ is the new surface point, $\vec{P}$ is the current surface point, $\vec{N}n$ is the normalized normal and the width is the displacement distance.

As a result object A displaced gradually and we got the variation of line thickness (see Fig. 10)

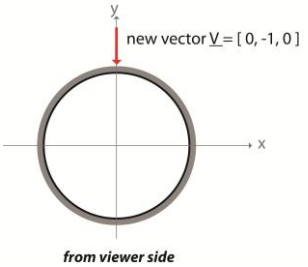


Figure9. The New Vector

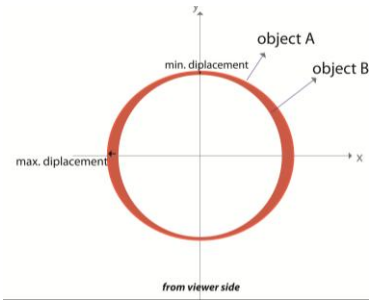


Figure10. Displacement ramp

4.3. Creating Wiggliness

When the artist paints the wayang beber of Pacitan, a small irregular movement of his hand causes the outline to be more or less wiggly. We displaced the surface point once more using a random number to resemble this wiggliness.

We generated the random number using the Perlin noise function. This noise function generates values which can be used to create randomness on a surface [20]. The noise function can take a varying number of parameters, and return a range of types. The values returned are guaranteed to be between 0 and 1. Nevertheless, in practice the output values are generally in the range of 0.27 to 0.7.

We applied the Perlin noise function to the texture (s and t) coordinates. The default values of s and t generated a low noise frequency. Therefore we scaled the values by multiplying them with a new parameter called *W*. Then to control the amplitude of the noise, we scaled the noise function output by multiplying it with a new parameter called *K*. Each surface point was displaced by the output of this noise function. As a result we achieved a wiggly outline (see Fig.11)

5. Discussion

The algorithm of our shader can generate the wayang beber of Pacitan outline. This shader simulates the incorrectness qualities of a human drawing.

We added some parameters to control several attributes of the outline, such as the thickness, the variation of thickness, and the wiggliness.

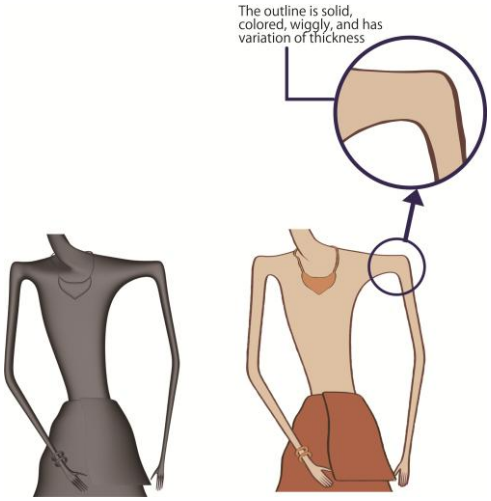
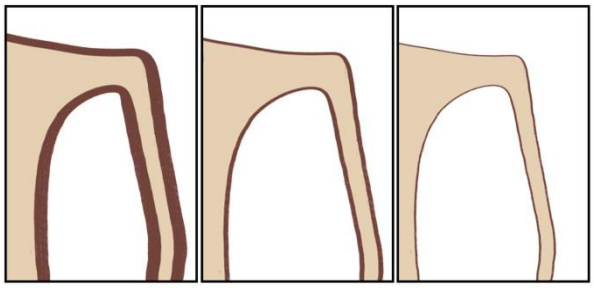


Figure11. The result of our shader

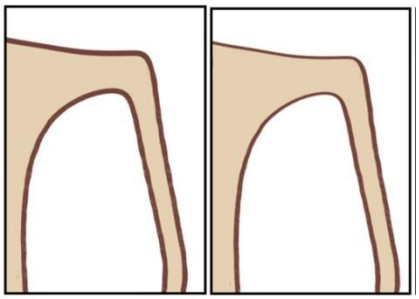
We rendered the object several times using different thickness values to get the good result. A comparison of the rendering result with different thickness values can be seen in Fig.12. The Fig 12a was obtained by thickness value of 0.3; the Fig.12b was obtained by thickness value of 0.1; and the Fig 12c was obtained by thickness value of 0.05. The desired result is Fig.12b.



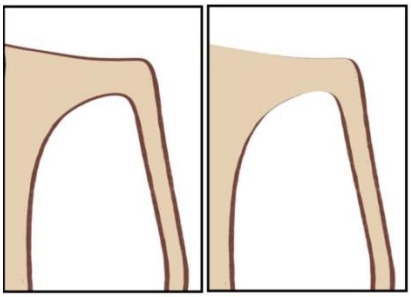
(a) **(b)** **(c)**

Figure12. Comparison of rendering result using different thickness parameter

We rendered the object several times to get a good balance of variations of thickness. The compared result can be seen in Fig.13.



(a) **(b)**

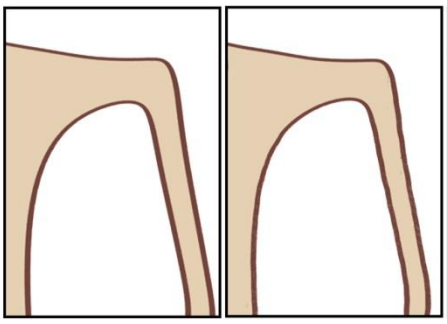


(c) **(d)**

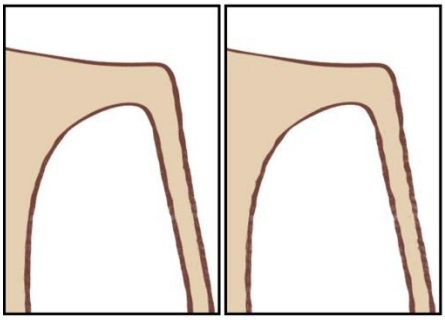
Figure13. Comparison of rendering result using different thickness variation parameter

The outline with uniform thickness can be seen in figure 13a which is obtained by a thickness variation value of 1. Fig.13b was obtained by the thickness variation value of 0.1. Fig.13c was obtained by the thickness variation value of 0.005. Fig.13d was obtained by the thickness variation value 0. The most appropriate result is the Fig.13b.

We also did an experiment using different values of wigglimess. The compared results can be seen in Fig.14.



(a) **(b)**



(c) **(d)**

Figure14. Comparison of rendering result using different wiggliness parameter

Fig.14a was obtained by the wiggliness value of 0. Fig.14b was obtained by the wiggliness value of 0.6. Fig.14c was obtained by the wiggliness value of 1. Fig.14d obtained by the wiggliness value of 1.5 . The best result is Fig 12b.

The comparison of our desired rendering result and the reference image can be seen in Fig.15.

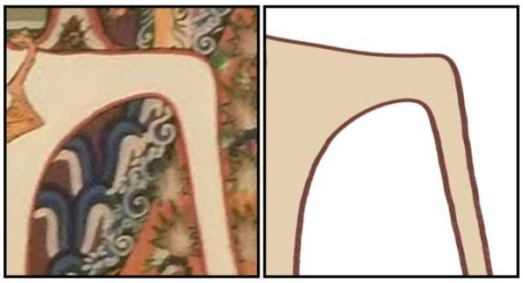
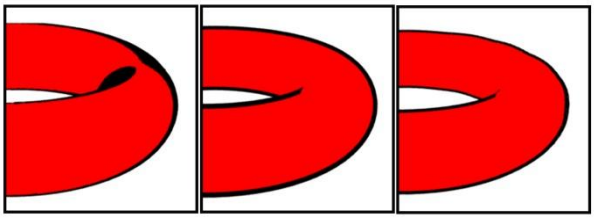


Figure15. Comparison between the reference image (left) and the rendering result (right)

We compared the rendering result of our shader with the result from another shader that was also developed using renderman. The comparison can be seen in Fig.16.



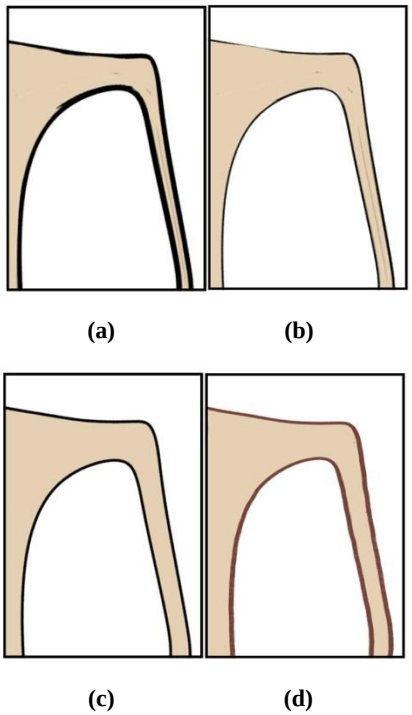
(a) **(b)** **(c)**

Figure16. Comparison of rendering result using different shaders. (a) Apocada and Gritz's shader, (b) ID outline shader, (c) our shader.

The shader developed by Anthony Apocada and Larry Gritz (Fig.16a) generates an outline with a variation of thickness. However, it also produced an unwanted shadow or dark area in the high curved edge. The contrast between the thick part and the thin part of the line is also difficult to control.

The shader developed by Ivan De Wolf (Fig.16b) generates a smooth thick outline. The rendering result is good, but not suitable for our purpose. Our shader (Fig.16c) tries to solve this problem. Our shader generates a relatively thick line with a variation of thickness and wiggliness which are suitable for our purpose.

Another comparison between the rendering result of these three shader can be seen in Fig.17.



(a) **(b)** **(c)** **(d)**

Figure17. Comparison of rendering result using different shaders. (a),(b) Apocada and Gritz's shader, (c) ID outline shader, (d) our shader.

6. Conclusion

The final goal of our research is to make CG animated wayang based on the visual form of wayang beber of Pacitan characters. This paper is the first step of the process. In this paper, we explained two things: first, the wayang beber of Pacitan outline features, and second, our algorithm to develop the outline shader.

Wayang beber of Pacitan is drawn by a drawing technique called "blabar". This technique created typical outline attributes, such as:

- thick
- variation of thickness.
- wiggliness.

These three attributes can be identified as incorrectness qualities of human drawing.

To simulate these qualities we developed our shader algorithm based on Ivan De Wolf's ID outline algorithm. We used a displacement ramp function to get the parameter to displace the surface point gradually. As a result we got a variation of line thickness. We displaced the surface point using a random number that was obtained by applying the Perlin noise function to the texture (s and t) coordinate. As a result we got a wiggly outline.

Our algorithm was implemented in the Renderman shading language (RSL). However there are some limitations in this algorithm. First, our algorithms produce a good result for rendering a curved object, but some problems occur when

rendering an object with sharp edges.

Second, our algorithm can easily control the outline attribute such as line width, variations of thickness, and wiggliness. However it is difficult to create a good balance between the object size and the outline attributes. We control the attributes parameter using an arbitrary number, but sometimes the resulting outline can be too thick or thin

7. Future Works

There are still many things to do to reach our goal. Our outline shader algorithm also has some limitation. It needs more improvement in the future.

There are several features of the wayang beber of Pacitan visual form that haven't been explained in this paper, such as the detail lines, colors, patterns, and textures. The next research should be able to explore all of these features, and develop algorithms to generate them using CG technology.

Rendering is just one stage of CG animation production. The animation process itself has its own problem. We intend to translate a 2D picture into 3D graphic animation. Our work has some similarities with Chan and Chen's [21]. Using the computer medium, we add a third dimension to the world of wayang beber of Pacitan paintings, which until now have traditionally been restricted to two dimensions. This is a new visual aesthetic different from the original one. Certainly, some problem will occur related to the shape deformations. Some 2D images have a distorted shape that will be difficult to simulate in 3D CG animation from a certain angle. We have found an approach to solve the problem using an algorithm to create a view-dependent model [22]. The view-dependent model is 3D CG models that will deform the shape depending on the viewer's current position. However, this may need more specific research.

References

[1] Mulyono, S., "Wayang; Asal Usul, Filsafat, dan Masa Depannya". Penerbit Alda, pp.53,1975.

[2] Van Dijk, K., Nas,P., *"Dakwah and indigenous culture; The dissemination of Islam"*. Bijdragen tot de Taal-, Land- en Volkenkunde, Globalization, localization and Indonesia, Vol.15, No.2, pp 224, 1998.

[3] Adisasmito, N. D., *Karakter Visual dan Gaya Ilustrasi Naskah Lama di Jawa Periode 1800-1920*, ITB Journal of Visual Art and Design, Vol. 2, No. 1, pp 64, 2008.

[4] Tabrani, P., *"Meninjau Bahasa Rupa Wayang Beber Jaka Kembang Kuning dari Telaah Cara Wimba dan Tata Ungkapan Bahasa Rupa Media Rupa Rungu Dwimatra Statis Modern dalam Hubungannya dengan Bahasa Rupa Gambar Prasejarah, Primitif, Gambar Anak dan Relief Cerita Lalita Vistara Borobudur"*, unpublished Ph.D. Dissertation, pp. 208, 1991.

[5] Cavalier, S., *The World History of Animation*, University of California Press, pp. 34, 2011.

[6] Curtis, C., *Non–photorealistic Animation,* SIGGRAPH 1999 Course no.17: Non-photorealistic Rendering, pp 3, 1999.

[7]Salmagundi, M., Magpie Salmagundi Personal Webpage, http://www.magpiesalmagundi.com/2012/03/wayang-beber.html, Accessed 12/Jul/2012.

[8] Salim : Warna Sunggingan dan Komposisi Wayang Beber Pacitan. Canthing Journal, Vol.1, No.1, pp. 1-9, 2012

[9] Ahmadi, A., *Identifikasi Pola dan Sunggingan Wayang Kulit Purwa Gaya Surakarta*, Research report, STSI Surakarta, p.206, 1997.

[10] Purbasari, T., *"Kajian Aspek Teknis, Estetis, dan Simbolis Warna Wayang Kulit Karya Perajin Wayang Desa Tunahan Kabupaten Jepara."*, Arty Journal of Visual Art, vol. 1, No.1, pp 1-7, 2012.

[11] Sukir, *Bab Natah Sarta Nyungging Ringgit Wacucal*, Departemen Pendidikan dan Kebudayaan, p.65, 1980.

[12]Grahita, B., Komma,T., Kushiyama, K., "Visual Analysis of Wayang Beber Pacitan Character Figure", JSGS 2012 conference proceeding, pp.71-74, 2012.

[13] Strohotte, T., Schlechtweg, S., *Non-phororealistic Computer Graphic*, Elsevier, pp84-85, 2000.

[14]Hajagos, B.,Szecsi, L., Csebfalvi, B., *Fast Silhouette and Crease Edge Synthesis with Geometry Shaders*, Proceedings of the 28th Spring Conference on Computer Graphics, ACM,pp 71-76,2012.

[15] Wolf, I. D., Ivan De Wolf Shaders PAGE, http://www.renderman.org/RMR/Shaders/IDShaders/index.html, accessed 2/Aug/2012.

[16]Apocada, A. A., Gritz, L., Andvanced Renderman Creating CGI for Motion Picture, San Diego: Academic Press, pp. 329-331, 2000.

[17]Gooch, B., Sloan, P., Gooch, A., Shirley, P. S., Riesenfeld, R., *Interactive Technical Illustration*, 1999 ACM Symposium on Interactive 3D Graphics, ACM SIGGRAPH , pp 31-38, 1999.

[18] Raskar, R., Cohen, M., *Image Precision Silhouette Edges*, 1999 ACM Symposium on Interactive 3D Graphics, ACM SIGGRAPH, pp 135-140, 1999.

[19] Apocada, A. A., Gritz, L., *Andvanced Renderman Creating CGI for Motion Picture*, Academic Press, pp. 22, 2000.

[20] Stephenson, I., *Essential Renderman*, Springer, pp 201-203, 2007.

[21]Chan, C., Akleman,E., Chen,J., *Two Methods for Creating Chinese Painting*, In Computer Graphics and Applications, Proceedings. 10th Pacific Conference on IEEE, pp. 403-412. 2002.

[22] Strohotte, T., Schlechtweg, S., *Non-phororealistic Computer Graphic*, Elsevier, pp299, 2000.