International Journal of Multimedia and Ubiquitous Engineering
Vol. 8, No. 1, January, 2013

Drawing Style Capture for Cartoon Rendering

Hae Won Byun* and Hye Moon Jung
School of Information Technology, Sungshin Women's University
{hyewon, hyemoon}@sungshin.ac.kr

Abstract

This paper proposes a method for capturing the drawing style of cartoon from the original comic book. Given a concept drawing image of the specific comic book, our approach recovers not only the color scheme, but also the color ratio. The main technical contribution of this paper is a shading model capture and a stylized cartoon rendering. For the shading model capture, we collect RGB statistics from the original drawing and sort the RGB data using its frequency and extract the shading model using dynamic threshold. We also propose the stylized cartoon rendering to represent 3d object using 2D texture according to the original cartoon style. We show on various cartoons that our approach can capture the drawing styles even in the case of rapid color changes and wide color schemes.

Keywords: *Cartoon Rendering, Drawing Style Capture, Non-photorealistic Rendering*

1. Introduction

Cartoon style rendering is a representative research field of non-photorealisitc rendering (NPR) techniques. The objective is to represent 3D object as a cartoon style. In this area, the focus is to simulate handmade illustrations with computer algorithms. This technique expresses the 3D object as a hand-drawn feel, such as pen and ink drawings, paintings or comic style renderings. This should appeal to the analog emotions, so, there are actively used in the several fields such as 3d animation, video game, and commercial advertising. Figure 1 shows these cartoon rendering examples.

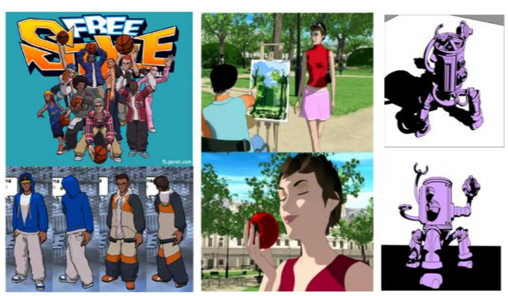


Figure 1. Cartoon Rendering Examples

Then, consider the process of making a comic book offline. The original comic book is produced by the multi-step processes such as concept setting, color setting and concept drawing. In this process, the concept drawing is an important basis for the final comic book. For example, in the cases of character production, 3D character model is made and is texture mapped based on the concept drawing. The Figure 2 shows the 3D character making process. In the shading stage, the animators undergo series of trials and error to imitate concept drawing style. It is difficult to render the 3D character by observing the 2D concept drawing and tedious to infer the color change and ratio of each region such as face, leg, and torso. To render the 3D character as a concept drawing style, the animator makes a texture to reflect the original feel of the picture. Because these texturing process is not straightforward and complex, the intuitive user interface is needed to express the shading of 3D object based on the concept drawing.

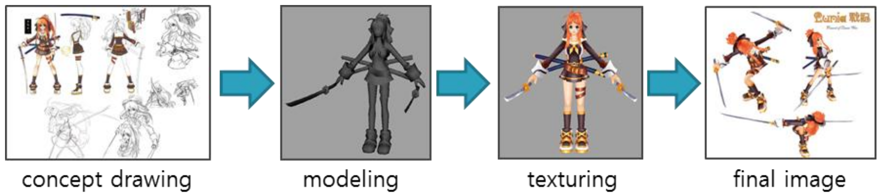


Figure 2. 3D Character Creation Process

In this paper, we present the drawing style capture method for the cartoon rendering. Our technology consists of two steps, the shading model capture and cartoon rendering as the concept drawing style. In the shading model capture step, using image segmentation, we divide the image into areas and select a desired area using a sketch interface. From the selected area, we capture the shading model by RGB color sorting algorithm based on dynamic threshold. Next, in the cartoon rendering step, we collect various cartoons and analyze the characteristics of the original cartoons. Adopting the 2D texture, we display 3d objects as a cartoon style and also represents the various effects, such as silhouette and highlight effects. Our technology enables cartoons are expressed as a variety of styles and can be applied to the various digital contents area such as animation, game, and movie.

We discuss previous work in Section 2. In Section 3, we show how to capture concept drawing style using RGB color sorting algorithm. In Section 4, we describe the cartoon rendering method based on 2D texture. Section 5 shows the experimental results of each drawing style which is captured from the original comic book. We conclude and discuss future directions of our work in Section 6.

2. Related Work

Our shading algorithm belongs to non-photorealistic rendering(NPR) field. Lake, *et al.*, [1], who propose the first cartoon rendering method, presented the method to represent 3D object as a cartoon style. Since then, his method has been applied and the research for the various stylization of cartoon rendering has been conducted. Barla, *et al.*, [2] extend 1D texture to 2D texture in order to represent view effect to be difficult of impossible to achieve with previous approaches. The x-axis of 2D texture represents the Lambertian equation and the y-axis reflects the shading factor such as z depth, distance to the outline, and reflection vector. Using this 2D texture, the object can be stylized as the level of detail(LOD) effect, the silhouette effect and highlight effect.

Spindler, *et al.*, [3] study on the shading style of the real 2D animation. By analyzing the shading style of the animation ‘Sin City’ and ‘Spawn’, he presented the stylized shadow, double contour lines, and soft cell shading. DeCoro, *et al.*, [4] research on the stylized shadow using the parameters of shadow inflation, brightness, softness and abstraction. This is inspired by the mere shadow of a real cartoonist.

To represent the 3D objects in the style of real art works, Sloan, *et al.*, [5] studied on capturing the shading model of the art works. He was inspired that he shades on the spherical objects, before real artists draw objects. By capturing the shading model of spherical objects from real art works, he shall make lit sphere to be applied to the 3D object. Kulla, *et al.*, [6] extract the color trajectory and texture material from the input of pain sample with RGB sorting algorithm. Yan, *et al.*, [7] present the automatic method of painterly art map(PAM) using simple sketch interface to express the unique feel of the brush touch in the art works. He also proposes the real-time rendering method to maintain temporal coherence between animation frames.

3. Shading Model Capture

Shading model capture is to extract the color change and ratio of the original concept drawing. We extract the shading model from the each region of the full picture. For example, we can extract the drawing style of the each region in the original image, such as grass, building and clothing. Therefore, we offer the simple user interface to select a specific area in the image. Using the user interface, the user can select the desired region to extract the shading model from the concept drawing. After selecting one particular area, we sort the color data of that area using RGB sorting algorithm and automatically make a ‘base texture’. To extract the highlight and silhouette effect, we select the particular area to extract that effect from the concept drawing. Then, automatically a ‘effect texture’ is generated. The final result texture is made by combining both the base texture and effect texture. The Figure 3 shows the extraction process of the shading model based on concept drawing.

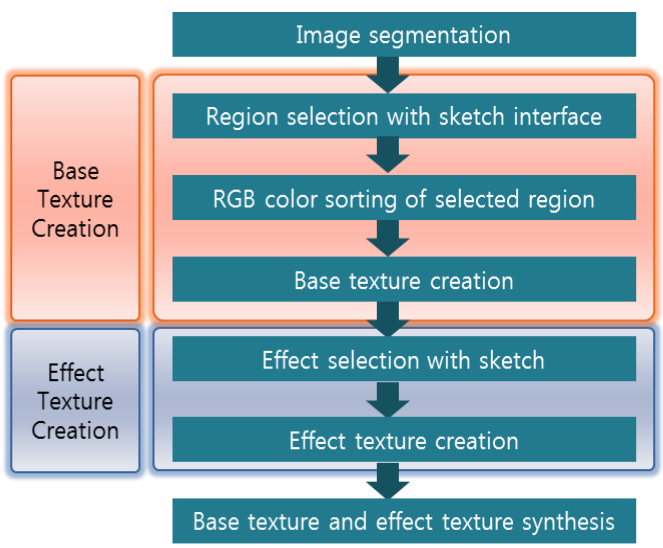


Figure 3. Shading Model Extraction based on the Concept Drawing

3.1 Image Segmentation

We use the Felzenszwalb’s algorithm [8] for image segmentation. The algorithm is to divide the region in the image according to the local color contrast and edge. Using the image segmentation algorithm, we can split the area and the user can select the desired region in the image using only the behavior of a simple sketch. Figure 4 shows the result of image segmentation.

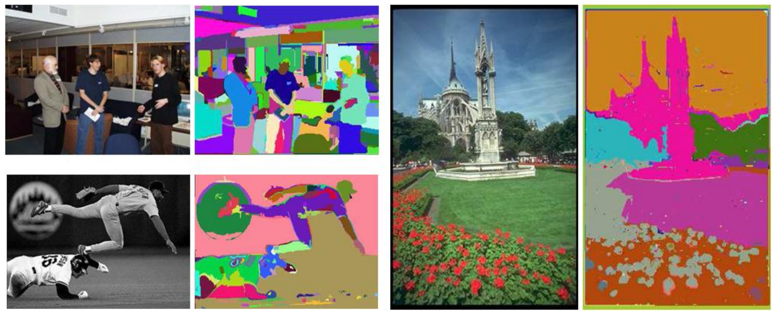


Figure 4. Image Segmentation Algorithm Results

3.2 RGB Sorting Algorithm

Generally, one of the drawing styles can be the color style to reflect the RGB color distribution characteristics of an image. We can extract the color style using the RGB sorting algorithm, presented by Kulla [6]. He extracts the color gradient and texture material from the input paint sample and then applies the method to express the shading of 3D object. In the extraction process, by independently extracting color changes and texture material from the paint sample, it enables texture reuse by applying the different color to the extracted texture. The extracted color from the paint sample is very diverse and not yet aligned. Therefore, Kulla suppose the RGB sorting algorithm in order to sort the color distribution. The input image of the algorithm is long horizontal 2D image and is increasingly lighter or darker from left to right.

Kulla's RGB sorting algorithm consists of 5 steps. The first step is to make a one dimensional color distribution by calculating the average value of all the columns of pixels in the pain sample. The second step is to start the algorithm in the both end points A and B of the unaligned color distribution. The third step is to find the central point M, given two points A and B based on RGB color information. The fourth step is to find the nearest point C with the point M. The fifth step is to insert the specific point to a linked list. If we find the point C, we perform the recursion algorithm between A-C interval and C-B interval. If we don't find the point C, we insert the A-B interval to a linked list by looking on the A-B interval to the nearest color.

By applying Kulla's sorting algorithm, the sorting result of color distribution is shown as Figure 5. That algorithm includes two limitations. The first, the input image have to be brighter from left to right because that algorithm starts at both the end points of the unaligned color pixels. The algorithm is performed correctly only if the constraints for the input image are satisfied up. Table 1 show both the correct result of the Kulla's algorithm and the incorrect result when the condition for the input image is not satisfied up. The second, the result closest to the straight line based on the first starting point of the algorithm is shown because the method is to find the C contained in the circle with the diameter A-B. For this reason, although the colors to occupy many areas from the input image, the color RGB space distance away is likely to be ignored. As you can see the Table 1, by observing the output color in the case of the incorrect result, you can find that the color RGB space away is ignored based on the start and end points of the input image.

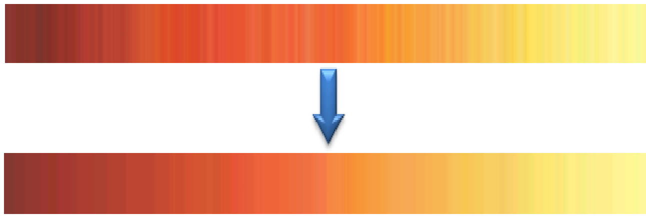
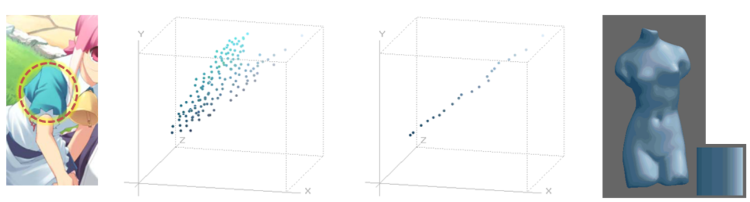


Figure 5. The Result of RGB Sorting Algorithm

Table 1. The Restriction of Kulla's Algorithm

	Input	*Output*
Correct result		
Incorrect result		
		
		

By adopting the Kulla's sorting algorithm, we try to solve the problem of drawing style capture from the original cartoon image. As you can see the Figure 6, (a) is the original cartoon image and (b) represents the color distribution of the image (a). The result of the sorting algorithm is shown in (c) of Figure 6. We do not make input image to meet the assumption of Kull's algorithm. Instead, we compute the color distribution form the original input. Therefore, we select the closest color to the black and the closest color to the white as the two starting point of the algorithm. As the result of this process, we got the wrong result. As you can see the (d) of Figure 6, the torso is painted mostly dark blue, but the selected area of the original cartoon is painted bright blue.



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

Figure 6. The Result of Kulla's RGB Sorting Algorithm

In order to overcome limitation, we present new RGB sorting algorithm based on color frequency. Instead color value itself, we consider the color frequency. The color frequency represents the number of color to occupy an area from the input image. Our algorithm allows that the color with the frequency to be higher than the threshold should not be ignored. Whereas Kulla's algorithm has the linear result of color sorting, our algorithm supports the non-linear result of color sorting. The (a) of the Figure 7 shows the result of the color frequency from the selected area of (a) of the Figure 6. Nearer the red color in (a) of the Figure 7 represents the higher frequency to correspond to the yellow area of the graph in (b).

First, the high frequency of the color is stored in a linked list and then between the colors is arranged with our sorting algorithm. This algorithm overcomes the constraints that the input image is brighter from left to right and the colors from the brightest color and the darkest color is not included in the result. Figure 8 shows a comparison of Kulla's sorting algorithm(a) and our algorithm(b). As you can see the Figure 8, (b) represents the more appropriate color compared to (a)

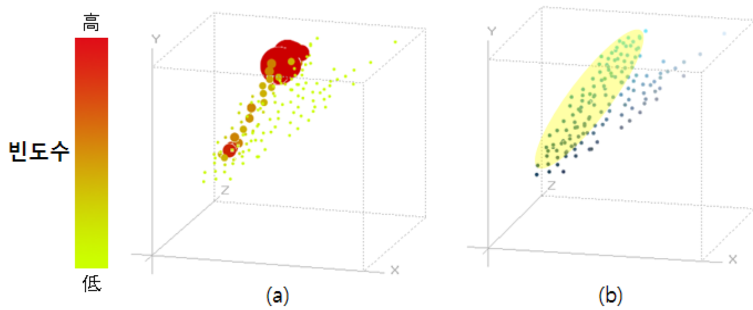


Figure 7. The Calculation Result of the Color Frequency of the Selected Area

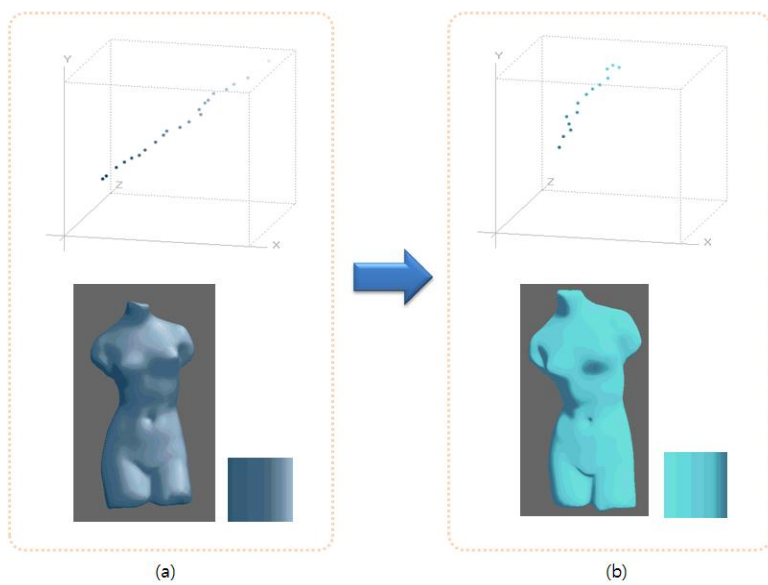


Figure 8. The Comparison of Results between the Kulla's Algorithm and the Frequency based Algorithm

4.3 Dynamic Threshold

To extract the color value with high frequency, it is important to determine the appropriate threshold. If the threshold is always the same regardless of the various cartoon types, it may cause less accurate problem because it applies the same threshold without taking into account the unique characteristics of each area. In order to solve the fixed threshold value problem, we adopt dynamic threshold to set the threshold dynamically according to the statistical characteristics of the area [9]. At this time, variance increases, the threshold would be increased. By using the variance of the color frequency, we may be able to determine the threshold to extract color values with the high frequency. When calculating the threshold, the influence of the average value can be ignored because the mean value is generally very small compared to the variance value. Therefore, the variance values can be thought of as the most important factor to determine the threshold. If we can formulate a first order linear equation that reflects the relationship between the variance distribution of the color frequency and the optimal threshold, we can estimate the dynamic threshold from the variance of the color frequency. Equation (1) is expressed by the formula.

$$T = a(\sigma^2)+b \tag{1}$$

Here, T is the threshold, σ^2 is the variance. Figure 9 shows the graph of the equation (1). The unknown a and b in this equation can be obtained by using the experimentally determined optimal threshold and the variance of the color frequencies. Through the experiments, we can formulate the linear first-order equation (2) to calculate the dynamic threshold. The optimal threshold can be statistically calculated by using the equation (2) .

$$T_{dynamic} = 0.0001(\sigma^2) + 20.852 \tag{2}$$

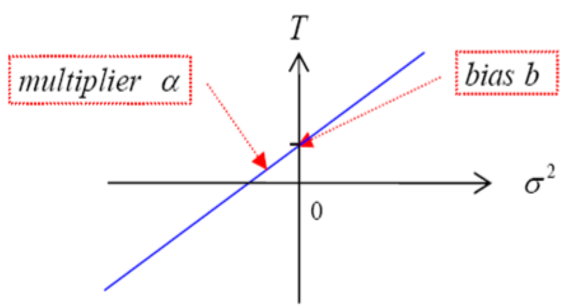


Figure 9. Dynamic Threshold Graph

4. Stylized Cartoon Rendering

For the purpose of the cartoon rendering, we collect the various cartoons and analyze the shading model of the cartoon. Based on the results of analysis, we present a new rendering method to represent the 3D object according to the shading style captured by the original cartoons.

By default, the cartoon rendering use the Lambertian shading model to determine the degree of the bright and dark color of each vertex using one dimensional texture. The shading value of 3D object is calculated by the dot product value of the light vector and the normal vector of each vertex. The higher the dot product value, the vertex facing the light source, the corresponding vertex of 3D object is expressed brighter. Vice versa, the vertex of object is expressed darker.

In this paper, we implement the cartoon rendering method using X-Toon method presented by Barla [2]. He proposed 2D texture that overcomes the 1D texture presented by Lake [1] to be difficult to represent the view effect of cartoon rendering. The x axis of the 2D texture reflects the Lambertian equation, the y axis represent camera view position, view direction or reflection vector. The view position can calculate the distance between the vertex and the camera position. The normalized value of the minimum distance and the maximum distance is applied to the y-axis of the 2D texture. The view direction is applied to the y-axis of the 2D texture as the dot product value of the normal vector and the direction vector of the camera. The reflection vector is applied to the y-axis as the dot product value of the direction vector and reflection vector of the camera. Through the extended texture, we can represent the level of detail(LOD) effect, silhouette effect, specular highlight effect to be appeared in the metal. The LOD effect use z depth information and the silhouette effect use the distance to the outline and the highlight effect use the reflection vector.

5. Experimental Results

We tested our drawing style capture method on the forty cartoons. Table 2 shows the area-specific variance and the optimal threshold and Max value means the maximum value of the frequency in that area. This table is expressed in a graph, as shown in Figure 10. Table 3 shows the results of equation (2) by calculating a dynamic threshold for each area.

Table 2. Area-specific Variance(σ^2) and the Optimal Threshold(Top)

region	variance	Threshold(T_{op})	Max	T_{op}/Max (%)
	66750.4	530	2012	26.3
	51913.5	350	1091	32.1
	40941.6	410	1451	28.1
	31787.4	300	1147	26.1
	3555.8	60	366	16.3
	3451.2	120	509	23.5
	2148.4	60	270	22.2

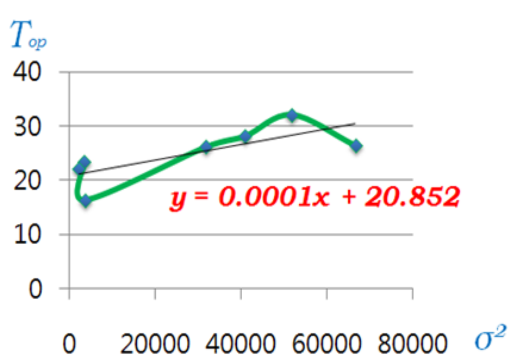
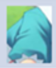


Figure 10. The Relationship between the Optimal Threshold Value and Variance

Table 3. Dynamic Threshold Value Calculation

region	variance	dynamic threshold
	66750.4	553.84
	51913.5	284.13
	40941.6	361.96
	31787.4	276.04
	3555.8	77.61
	4302.2	108.53
	2148.4	56.88

We tested our approach on the game ‘Kid Icarus’. In the Figure 11, (a) shows the shading model to be captured from the game and (b) displays the result to apply the shading model to the 3D model.

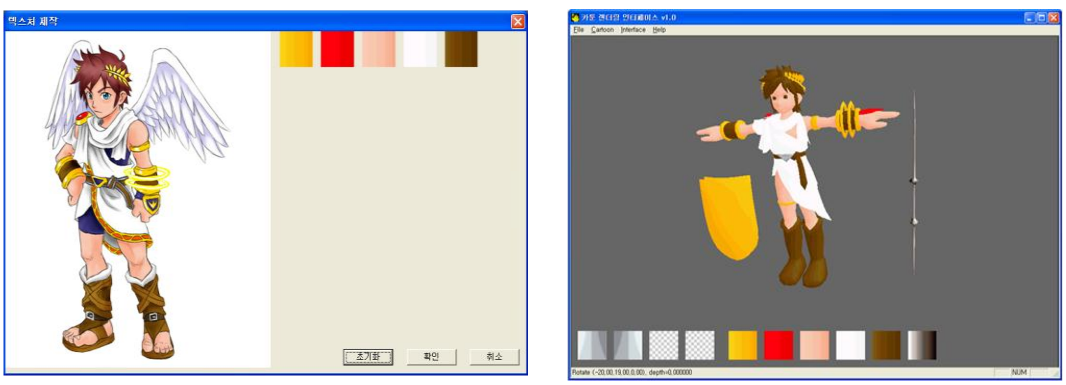


Figure 11. (left) Concept Drawing (right) Stylized Cartoon Rendering

6. Conclusion

We have presented the capturing method of various cartoon styles from the original comic book. Our approach lets the designer create a cartoon style contents easier and allows the designer to create the cartoon style according to the drawing style of the original cartoon book. Through the use of RGB color sorting, one can automatically extract the color scheme and the color change of the original cartoon. By adopting the 2D texture, a designer can easily create a wide range of effects(silhouette effect, highlight effect) in cartoon that were difficult to achieve with previous methods. At future work, we will also conduct a study on the capturing of cartoon-style outline, as well as shading.

Acknowledgements

This work was supported by the SungShin Women's University Research Grant of 2012.

References

[1] A. Lake, C. Marshall, M. Harris and M. Blackstein, "Stylized Rendering Techniques for Scalable Real-Time 3D Animation", Proceedings of SIGGRAPH, **(2000)**.
[2] P. Barla, J. Thollot and L. Markosian, "X-Toon: an Extended Toon Shader", Proceedings of the International Symposium on Non-Photorealistic Animation and Rendering (NPAR), **(2006)**.
[3] M. Spindler, N. Rober, R. Dohring and M. Masuch, "Enhanced Cartoon and Comic Rendering", Proceedings of EUROGRAPHICS, **(2006)**.
[4] C. DeCoro, F. Cole, A. Finkelstein and S. Rusinkiewicz, "Stylized Shadows", Proceedings of the International Symposium on Non-Photorealistic Animation and Rendering (NPAR), **(2007)**.
[5] P. J. Sloan, W. Martin, A. Gooch and B. Gooch, "The lit sphere: a model for capturing NPR shading from art", Proceedings of SIGGRAPH, **(2001)**.
[6] C. D. Kulla, J. D. Tucek, R. J. Bailey and C. M. Grimm, "Using Texture Synthesis for Non-Photorealistic Shading from Paint Samples", Proceedings of PACIFIC GRAPHICS, **(2003)**.
[7] C. R. Yan, M. Chi, T. Lee and W. Lin, "Stylized Rendering Using Samples of a Painted Image", IEEE Transaction on Visualization and Computer Graphics, **(2008)**.
[8] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Graph-Based Image Segmentation", International Journal of Computer Vision, **(2004)**.
[9] G. Taubin, "Estimating the tensor of curvature of a surface from a polyhedral approximation", IEEE Computer Society, **(1995)**.

Authors

Hae Won Byun

Hae Won Byun received the BS degree in computer science from Yonsei University, Korea, in 1990 and MS and PhD degrees in computer science from Korea Advanced Institute of Science and Technology (KAIST), Korea, in 1992 and 2004, respectively. She is currently a professor in the School of Information Technology at the Sungshin Women's University, Seoul, Korea. She's primary research interests are facial motion capture and expression retargeting, non-photorealistic rendering, and texture synthesis.

Hye Moon Jung

Hye Moon Jung received B.A. and M.S. degrees in school of Media & Information from Sungshin Women's University, in 2007 and 2009. Her current research interests are non-photorealistic rendering, game and HCI.