Automatic Pixel Art Generation from 3DCG Models

Nobutoshi SHIMIZU
Tokyo Denki University
Tokyo, Japan
n-shimizu@vcl.jp

Yuki MORIMOTO
Tokyo Denki University
Tokyo, Japan
yuki@vcl.jp

Tomoki MORIYA
Tokyo Denki University
Tokyo, Japan
moriya@vcl.jp

Tokiichiro TAKAHASHI
Tokyo Denki University, UEI Research
Tokyo, Japan
toki@vcl.jp

***Abstract*— We propose an automated method to generate pixel art from a 3DCG model in real time by specifying its resolution. Our method also emphasizes contours by darkening their colors. Pixel art is one of the popular styles that are used in games. However it takes time to design pixel art images because a 3DCG model can be seen from many directions and it should be rendered in pixel art style for all directions. Furthermore, we need more variation of pixel art images for animation. Expertise knowledge and techniques are also required for manually generating pixel art.**

***Key words* —3DCG, pixel art, contour detection.**

I. INTRODUCTION

Generating pixel art takes a long time and needs specialized knowledge and skills. Automated method to generate pixel art is needed in recent years as one of 3DCG rendering styles. For 3DCG, models are viewed from multi-direction. There are some effective methods to produce pixel art using 3DCG model as a rough design [1][2]. These approaches have two advantages. First, the designer can omit the process of rough sketching. Secondly, it eases individual variations of designer which leads to uniform quality.

In this paper, we present a method to generate a pixel art automatically from 3DCG model. Figure 1 shows an input 3D scene and pixel art generated by our method. Our result shows that our method enables to extract the suitable contours for pixel art from 3DCG model.

(a) Pixel art (b) Pixel art's contour

Figure 2: An example of pixel art and its contour.

(a) Input 3DCG model (b) Our result

Figure 1: Input 3DCG model and our result.

II. DEFINITION OF PIXEL ART

In this paper, we define features of pixel art from the textbooks of dot picture production [3][4][5] as follows.

- Emphasized contours
  Figure 2 shows a contour of the pixel art. Contours of pixel art are rendered as connected pixels in eight neighbors. Contours have darker color than the surrounding foreground.
- Limited number of colors
  Pixel art was devised for computers and game hard in 1980s. At that time, the number of colors for one object had been limited. In this paper, we limit the number of colors in the same way as 1980s.
- Low resolution, less than 512 × 512 pixels
  There is no definition about resolution of pixel art. In this paper, we define that pixel art's resolution is less than 512 × 512 pixels because such low resolution can often be seen in commercial game products. For example, the game [6] applies pixel art generated by 264 × 448 pixels for a character.

III. Previous research

Matsushima et al. [1] proposed a method to generate a contour of the pixel art from 3DCG model. They get the RGB information and depth information by rendering a 3DCG model at low resolution. First, they perform a Canny edge detector to obtained depth image to extract a contour. Then the contour of 3D model are obtained by a thinning process of the obtained contour. The contours are colored darker. For other parts, the corresponding colors to 3D model are obtained from the RGB image to render a contour. Their method composes the generated contour and the RGB images rendered in low resolution that result in a pixel art. Matsushima's method can draw the contour which forms a boundary between the background and the 3DCG model, but it cannot draw the contour on the inside 3DCG model.

Kopf et al. [7] proposed content-adaptive image downscaling. Their method maintain the detail of the input image and prevent aliasing that occurs when the downscaling. They claim that their method is effective in the generation of pixel art because the sharp edge can be maintained. However, such a 2D-based approach are depends only on the luminance information, If the adjacent area is similar colors, extracting contour and retaining features are difficult.

IV. Algorithm

In this section, we describe the algorithm that generates the pixel art defined in the previous section. Figure 3 is a rendering pipeline of our proposed method.

A. Toon Rendering

Gradual shading representation often appears in the pixel art. We obtain each object's toon rendered RGB image (Fig. 3 (b)) from input 3DCG model (Fig. 3 (a)).

B. Depth Rendering

In the process of a contour extraction from the RGB image, unintended color edges can be detected, whereas contours cannot be detected where the colors of the 3D model and the background are similar. We use the depth information to avoid such problems. We obtain depth images for each 3DCG object (Fig. 3 (a, d)).

C. Down Sampling

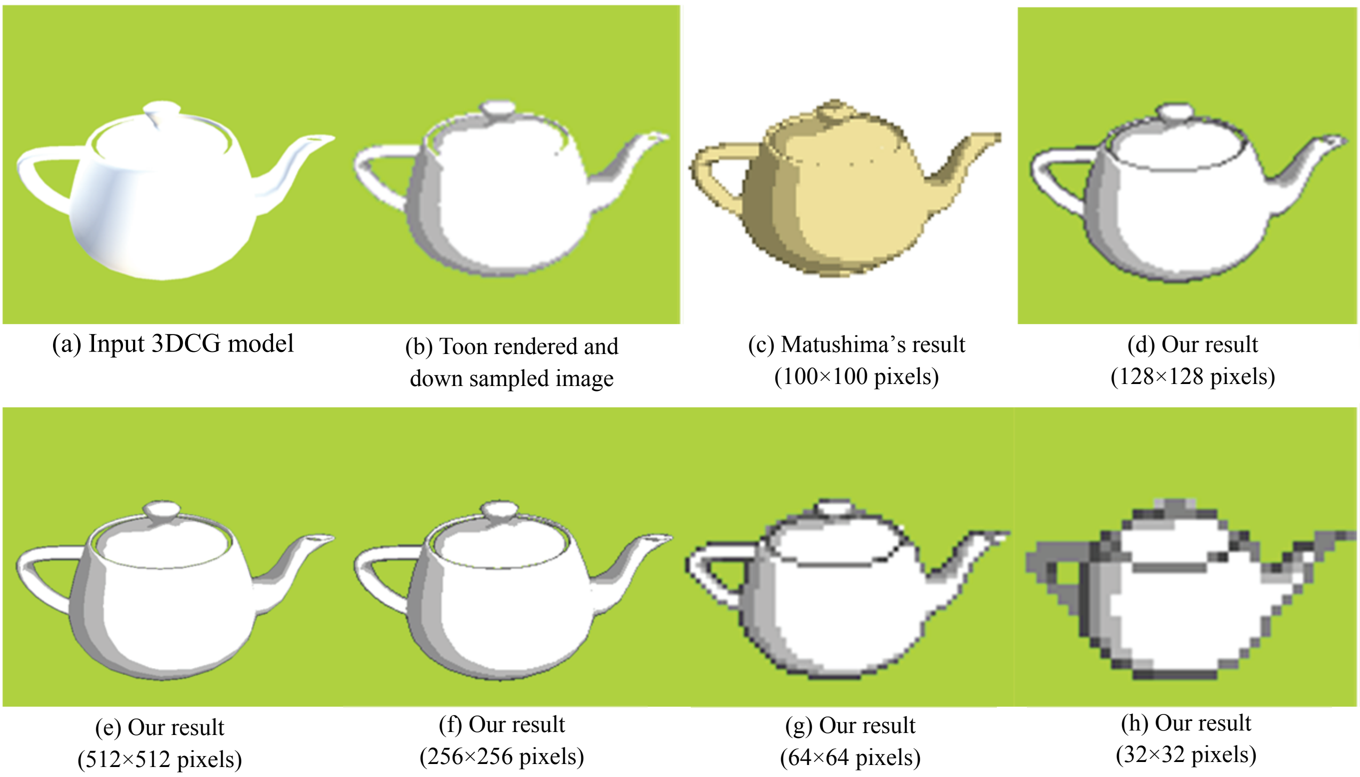
We obtain a low-resolution image (Fig. 3 (c)) of the RGB image (Fig. 3 (b)) which is downscaled in n × n intervals. The color of each downscaled pixel is calculated to correspond with the color in the RGB image at the pixel’s center position.

D. Contour Detection

We divide the depth image (Fig. 3 (d)) into blocks of n × n pixels. We compute the difference between the maximum and the minimum depth value in each block. If the difference is larger than an arbitrary threshold, the block is detected as a contour image (Fig. 3 (e)).

E. Thinning Processing

Pixel art's contour is rendered as 8-connected neighboring pixels. However, contour image's contour is a line of 4-connected. Therefore, we convert the 8-connected line to the 4-



(a) Input 3DCG model
(b) Toon rendered and down sampled image
(c) Matushima’s result (100×100 pixels)
(d) Our result (128×128 pixels)
(e) Our result (512×512 pixels)
(f) Our result (256×256 pixels)
(g) Our result (64×64 pixels)
(h) Our result (32×32 pixels)

Figure 5: Our result and comparison between our method and conventional method.

connected line by the thinning processing. We obtain a thinned contour image by the thinning process to a contour image.

F. Darkening Contour

We calculate the minimum depth value for each n × n pixel block to detect contours. We obtain the color at same position from the RGB image using the coordinates of the pixels obtained earlier. We convert the obtained color to HSV color space from the RGB color space. Then, we reduce the brightness by γ% to get a dark contour image (Fig. 3 (g)). We set $\gamma = 50$ for our results in this paper.

G. Compositing Low-Resolution Image and Dark Contour Image

We obtain a pixel art (Fig. 3 (h)) by composite of the low-resolution image (Fig. 3 (c)) and dark Contour image (Fig. 3 (g)).

H. Compositing All Objects

We obtain a generated pixel art (Fig. 3 (i)) by composite of each pixel art (Fig. 3 (h)) (Fig. 3 (h')). We apply depth buffering by considering the depth value of each object for this process.

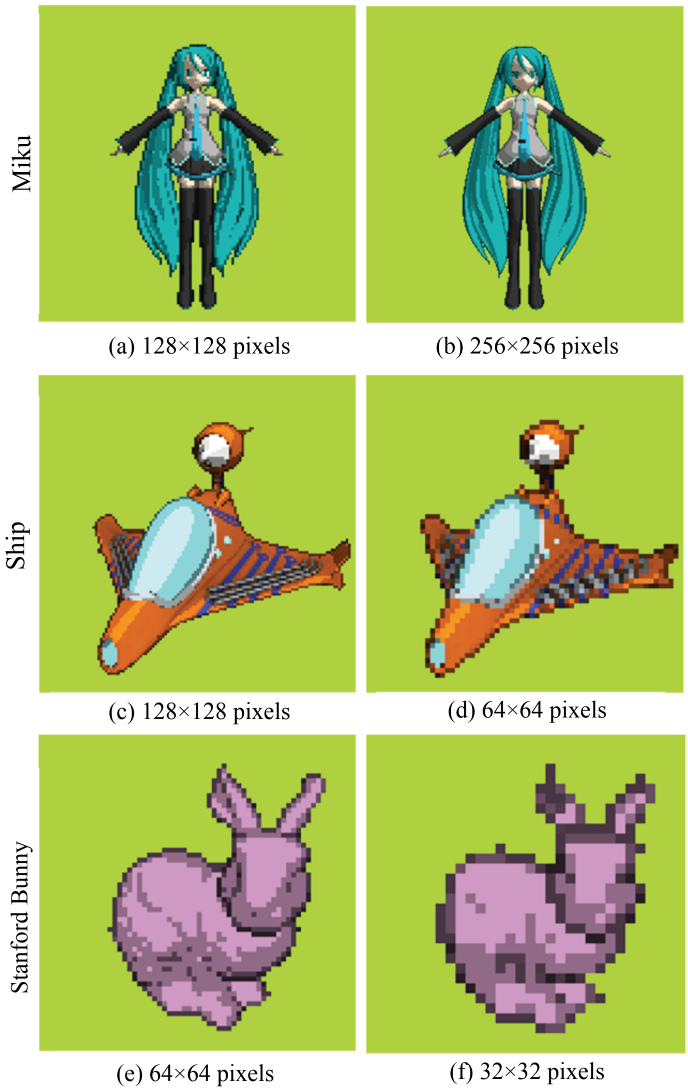


Figure 4: Generated pixel art images by our method.

V. RESULT

Figure 4 is a dot picture generated by our method. In our results, contours are connected in 8 neighboring pixels and have color darker than the surrounding color. Our method is also stable at any resolution.

Figure 5 shows a comparison between our results and Matsushima's results [1] and toon shading + down sampling results. In the comparison, our method can detect contours that these conventional methods miss.

From the results above, the proposed method is effective in generation of pixel art.

VI. CONCLUSION

In this paper, we have proposed a method to generate a pixel art automatically from 3DCG model. The proposed method was successful in detecting contours for the pixel art. Our pixel art generation algorithm does not require the specialized knowledge and skills of pixel art.

In the proposed method, we focused on the contour extraction. However, the feature of pixel art is not only contour. Color is also an important factor. In the proposed method, the color was determined by a simple toon and down sampling. We would like to improve the process of color reduction by the color palette of 3DCG model data.

In future work, we would like to consider geometric features of the 3DCG model for emphasizing colors.

REFERENCES

[1] SNK PLAYMORE CORPORATION, “Graphic, full of the soul and its creation process”, (tamashii no gurafikku sono seisakukoutei) http://kofaniv.snkplaymore.co.jp/info/15th_anniv/2d_dot/creation/ (2014.10.09 accessed) (in Japanese).

[2] Tatsuya Matsushima, and Taichi Watanabe, “The method of generating outline for character Pixel Art using 3D model”, ITE Tech. Report, 37(17), pp.31-34, 2013 (in Japanese).

[3] W. Miduho, and T. Nakamura, “Game Graphics Design - how to draw pixel art character (dottoekyara no egakikata)”, SB Creative, Japan, 2008 (in Japanese).

[4] A. Sabato, “Pixel art workshop - anyone can use pixel art technique is packed!!(dottoe koubou - daredemo tukaeru dottoe tekunikku ga mannsai!!)”, X-media, Japan,2006 (in Japanese).

[5] M. Nishimura, and R. Kitami, “Textbook of pixel art - from basic to craftsmanship (dottoe no kyoukasyo – kiso kara takuminowaza made-)”, Aspect Corporation, Japan, 2006 (in Japanese).

[6] ARK SYSTEM WORKS, “BLAZBLUE CHRONOPHANTASMA”, http://www.blazblue.jp/cp/ (2014.11.20 accessed) (in Japanese)

[7] Johannes Kopf, Ariel Shamir, Pieter Peers, “Content-adaptive image downscaling”, ACM Transactions on Graphics (TOG), Volume 32, Issue 6, November 2013, Article No. 173.

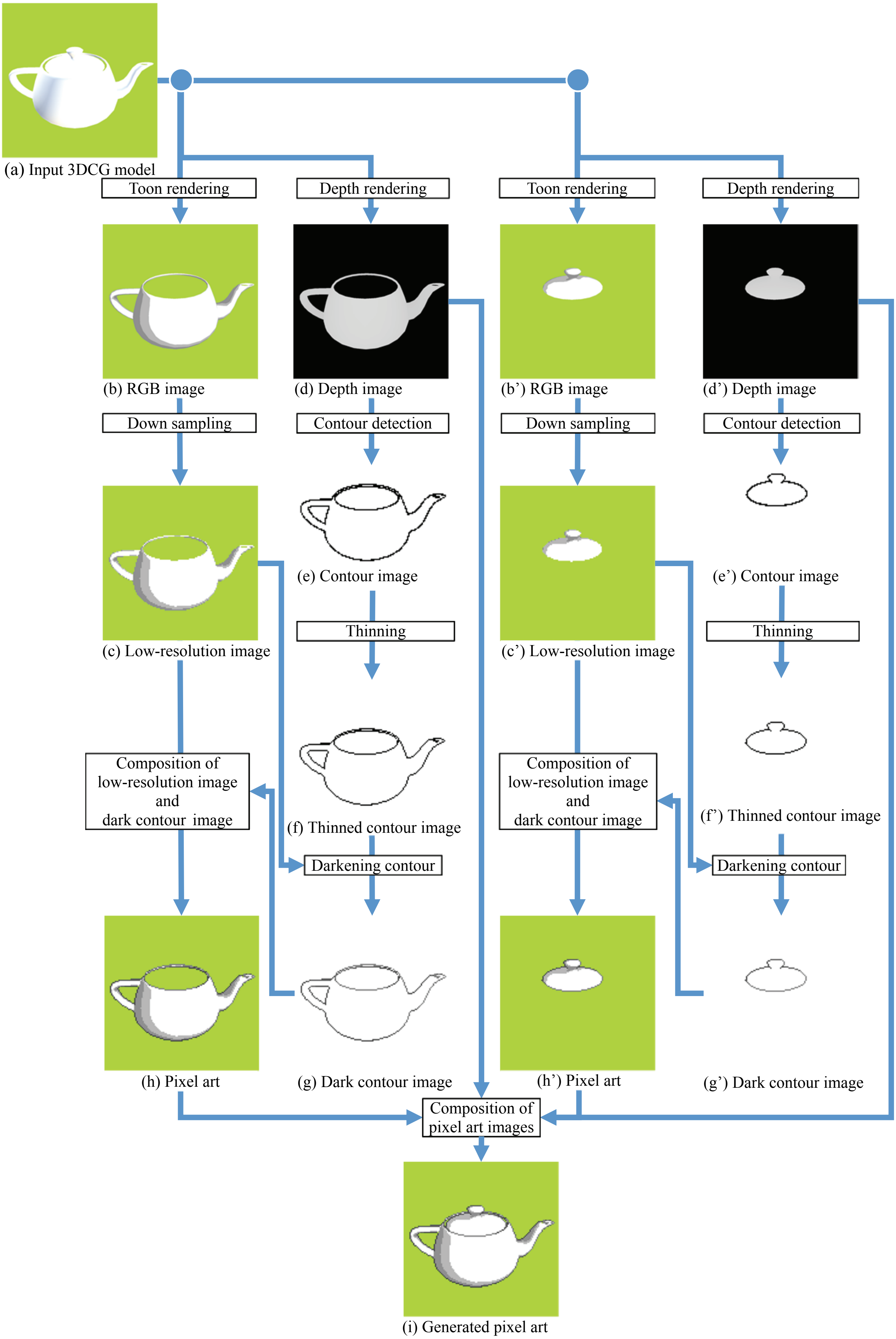


Figure 3 Rendering pipeline of our proposed method.