MULTICOLOR REMOVAL BASED ON COLOR LINES AND IMPROVED HOUGH TRANSFORM FOR SFS

Tianqi Wang[1] *and Terumasa AOKI*[1,2]

[1]Graduate School of Information Science(GSIS), Tohoku University
[2]New Industry Creation Hatchery Center(NICHe), Tohoku University

ABSTRACT

Constant albedo of surface is an important premise widely used for Shape from Shading(SFS). Since it is not satisfied when the surface is in multicolor, the application of SFS methods are very limited. In this paper we introduce a novel method for gray-scale transformation as a preprocess of SFS methods, which removes intensity differences between colors. First we cluster the pixels based on color using an improved Hough Transform approach, and generate Color Lines to describe color information. Then the color is adjusted using Color Lines, and the multicolor image is converted into gray-scale without sudden change of intensity between regions in different color. Experimental results show that our method outperforms existing methods such as linear luminance when 3D models are reconstructed by SFS.

Index Terms— Color Lines, SFS, Gray-scale, Hough Transform

1. INTRODUCTION

Shape From Shading, which reconstructs 3D models from one single input image using the shading information, is an ill posed problem[1]. There's been two excellent survey paper [2, 3] discussing the details of development of SFS methods. Besides, new methods are proposed in recent years[4][5][6][7][8]. As Shading information is extremely important for SFS methods, preprocessing methods are also proposed to extract better shading information to improve the results of SFS method[9].

To extract shading information from an RGB image correctly, gray-scale image is required as input. Linear luminance is a very popular technique to convert a multicolor image into gray-scale:

$$Y = 0.299R + 0.587G + 0.114B \tag{1}$$

where Y is the gray-scale intensity, and R/G/B are RGB values of each pixel. However the result of linear luminance remains the different albedo of colors, resulting in incorrect shading information. The usage of SFS is strongly limited for multicolor objects, hence a preprocessing is necessary to remove the difference between colors to make it feasible for SFS methods for multicolor objects. Few researches on multicolor removal have been done. M. Rahman et al[10] proposed a method based on Edge Flow Segmentation to merge regions in different colors into one single color. [11] uses chromaticity instead of Edge Flow Segmentation.

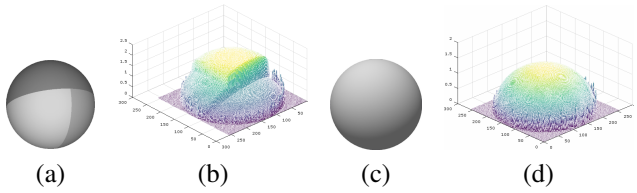

Fig. 1. Linear luminance can't remove the difference of albedo. (a) is gray-scale image of multicolor objects generated by linear luminance. (c) is an object in single color. (b) and (d) are results of SFS. The (a) can't be processed by SFS because the albedo of each color is different.

The first challenge of this task is color detection. Color Lines proposed in [12] describes color information using RGB histogram. Because of the change of lightness, the distribution of RGB values of multicolor image is in trend of lines in RGB space. Color with change of lightness can be represented by line model in RGB space, but the generation of Color Lines is not easy when the distribution of RGB points is complex. Hough Transform[13] is a way to help improve Color Lines' generation. However, it is extremlly time consuming when Hough Transform is applied in 3D space. We convert the 3D Hough Transform into three 2D Hough Transforms to solve this problem. Then we propose a method based on Color Line to adjust colors to convert the input image into gray-scale.

We cluster pixels to each color line based on the distance between pixels to color lines in RGB space. Then we calculate a novel model of adjustment for pixels belonging to each color line using the geometry properties of color lines in RGB space. This method transforms the problem of scalar value adjustment (intensity of each pixel) into the problem of geometry translation in RGB space. The output of our method is a gray-scale image with no differences between colors. We

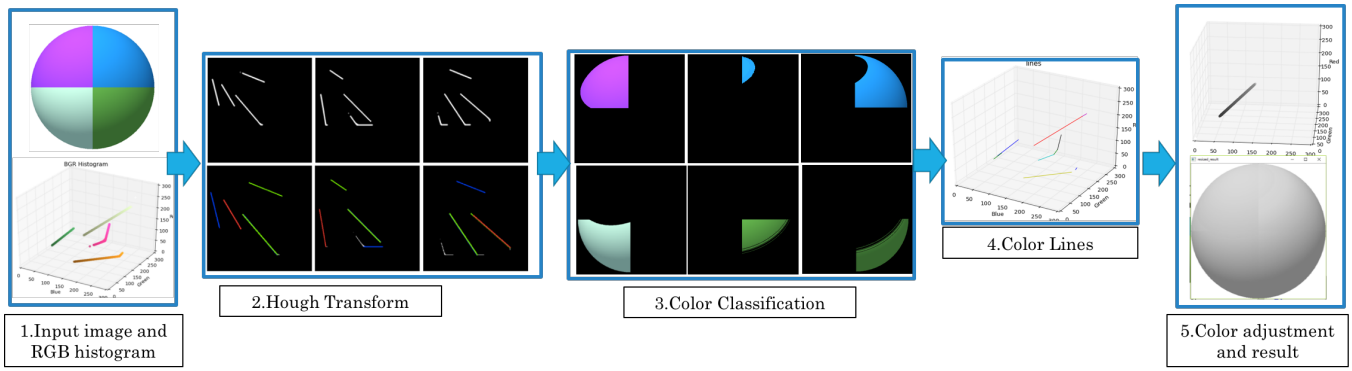

Fig. 2. Work flow of our method.

give an example of the work flow of the method in Fig.2.

The rest of this paper is organized as follows. Section 2 introduces the way we classify pixels into different colors using an improved Hough Transform in 3D space, and section 3 shows a novel method to adjust RGB values based on Color Lines model. Experimental results are shown and discussed in section 4, where we apply SFS methods to test the performance of our method. And we give the conclusions and future work in section 5.

2. CLASSIFICATION OF PIXELS' COLOR

As is shown in Fig.2 (1), for an object in multicolor, the distribution of RGB values in RGB space is approximately line-like because of the change of lightness on the surface with the change of surface's orientation. [12] proposed the Color Lines model to classify pixels in different colors, but we found that mistakes happen when lines are close to each other, because it's confusing how to connect color line points to generate the line model. To solve this problem, we cluster the pixels based on color first, and generate line model for each cluster of pixels one by one.

2.1. Color Classification

We detect and classify color of pixels using improved Hough Transform. Traditional Hough Transform finds the best linear parameters by voting, but it's very time consuming when we apply it in 3D space. Thus it's not feasible to generate line models for RGB histogram directly.

Fig.2 (2) shows the way we apply the Hough Transform. To avoid applying Hough Transform in 3D space directly, we project the RGB points to 3 planes of R-G, R-B, G-B, as is shown in the top row of Fig.2 (2).

On each plane, projected 2D points are distributed in lines. We apply 2D Hough Transform on each plane to generate lines for the projections, as is shown in the bottom row Fig.2(2). Then we cluster each projection point to one of the hough line depending Euclidean distance. For each RGB point $\mathbf{c}$, there are three projections $\mathbf{c_{rg}}, \mathbf{c_{rb}}, \mathbf{c_{gb}}$, each of them belonging to one hough line. As a result, pixels with RGB values of $\mathbf{c}$ is labeled by 3 hough lines, classifying it to one color. Fig.2 (3) shows the result of classification.

2.2. Generation of Color Lines

Since we have clustered the pixels based on color, we get start to generate Color Lines for each of the cluster. First we generate color line points using hemisphere. For each pixel $\mathbf{p}$ of an input image, we take the values of RGB channels r_p, g_p, b_p as the location of $\mathbf{p}$ in RGB space. Here, denote

$$n_p = \sqrt{r_p^2 + g_p^2 + b_p^2} \tag{2}$$

as the RGB norm of $\mathbf{p}$. Denote s as the number of slices, then the width of each slice is

$$w = n/s \tag{3}$$

$$n = n_{max} - n_{min} \tag{4}$$

where n_{max} and n_{min} are the maximum and minimum of RGB norms among a cluster $\mathbb{C}$. Intersection points of pixels and the i^{th} hemisphere h_i are defined as

$$\mathbb{P}_i = \{\mathbf{p}| \, |n_p - r_i| < 3\} \tag{5}$$

Cross sections are generated in this way, and one color line point is calculated in each cross section. Take $\mathbf{p_i^c}$ to denote the color line point on hemisphere h_i,

$$\mathbf{p_i^c} = \frac{\sum_0^n \mathbf{p_j}}{n}, \mathbf{p_j} \in \mathbb{P}_i \tag{6}$$

Color line skeletons are generated by connecting color line points, denoted as $(\mathbf{p_i^c}, \mathbf{p_j^c})$, where $\mathbf{p_i^c}$ and $\mathbf{p_j^c}$ are color line points. For skeleton $(\mathbf{p_1^c}, \mathbf{p_2^c})$ and the next color point $\mathbf{p_3^c}$, θ is the angle between $(\mathbf{p_1^c}, \mathbf{p_2^c})$ and $(\mathbf{p_2^c}, \mathbf{p_3^c})$. Update $(\mathbf{p_1^c}, \mathbf{p_2^c})$ with $(\mathbf{p_1^c}, \mathbf{p_3^c})$ if $cos\theta > 0.95$, else save $(\mathbf{p_1^c}, \mathbf{p_2^c})$ and generate new skeleton $(\mathbf{p_2^c}, \mathbf{p_3^c})$. The skeletons are color lines for

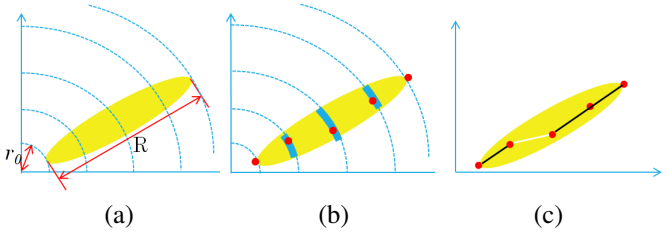

Fig. 3. Steps of generating Color Lines in RGB space (shown in 2D Figure).

$\mathbb{C}$. Finally, pixels are clustered to one of the Color Lines depending on the Euclidean distance between the RGB points of the pixels and the Color Lines.

Fig 3 shows the steps of generating Color Lines.The yellow region is RGB points clustered to same color cluster. The blue dotted lines are hemispheres. The green bold lines are intersection points. The red dots are color line points. The white and black lines are color line skeletons.

3. REMOVAL OF MULTICOLOR

This section introduces the details of our method to transform RGB points onto one standard line. It's easy to notice that the RGB distribution of a gray-scale image is in form of standard line l_s:

$$R = G = B \tag{7}$$

To transform a multicolor image into gray-scale, we construct a model M that for a pixel $\mathbf{p} = (x, y)$ with color of $\mathbf{c}(r, g, b)$:

$$M(\mathbf{c}) = (r', g', b') \tag{8}$$

$$(r', g', b') \in l_s \tag{9}$$

Fig.4 shows the transform model M_c of color line l_c : $t\mathbf{p^c} + (1-t)\mathbf{q^c}, t \in (0, 1)$, where $\mathbf{p^c}$ and $\mathbf{q^c}$ are the terminal points of l_c in RGB space, consisting of models of *translation*, *rotation* and *stretch*. The line in green is a color line, while the line in red is the standard line. We first translate l_c to $\mathbf{o}(0, 0, 0)$, rotate it to the direction of the standard line l_s, move it to position (n, n, n) where $3 * n^2 = |op_c|^2$, and adjust the length of it. We define M_c in the way of:

$$M_c(\mathbf{c}) = ((\mathbf{c} + \mathbf{T_{c1}}) \cdot \mathbf{R_c} + \mathbf{T_{c2}}) * k_c, \mathbf{c} \in l_c \tag{10}$$

$$\mathbf{T_{c1}} = -\mathbf{p^c} \tag{11}$$

$$\mathbf{T_{c2}} = (\frac{n_p}{\sqrt{3}}, \frac{n_p}{\sqrt{3}}, \frac{n_p}{\sqrt{3}}) \tag{12}$$

where R_c is a 3×3 rotation matrix that rotates l_c to direction of $(1, 1, 1)$. The parameter k_c is used for *stretch* to adjust the length of l_c.

For a pixel $\mathbf{p}$ of the input image, $\mathbf{q}$ is $\mathbf{p}$'s neighbor if

$$\mathbf{p} + \mathbf{v} = \mathbf{q}, \mathbf{v} \in \{(0, 1), (0, -1), (1, 0), (-1, 0)\} \tag{13}$$

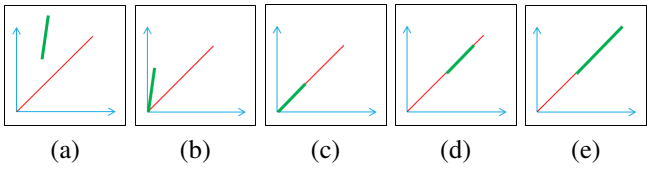

Fig. 4. Transform model. (a) is a color line(in green) l_c and the standard line(in red). (b) is to translate l_c to O(0, 0, 0). (c) is to rotate l_c. (d) is to translate l_c to (n, n, n). (e) is to adjust the length of l_c.

Edge $\mathbb{E}$ is defined as :

$$(\mathbf{p}, \mathbf{q}) \in \mathbb{E}_{12}\ if\ \mathbf{p} \in l_1, \mathbf{q} \in l_2\ and\ l_1 \neq l_2 \tag{14}$$

where $\mathbf{p}, \mathbf{q}$ are neighbors. To merge l_1 to l_2, parameter k is calculated by

$$k_{12} = \frac{1}{N} \sum_{(\mathbf{p},\mathbf{q}) \in E_{12}} \frac{n_q}{n_p} \tag{15}$$

where N is the cardinal number of $\mathbb{E}_{12}$, $\mathbf{n_p}$ and $\mathbf{n_q}$ are norm of $\mathbf{p}$ and $\mathbf{q}$.

Then we apply M_c to each RGB point $\mathbf{c} \in l_c$ to adjust the color. As the last step, to make sure all the points are moved onto the standard line, we apply linear luminance to obtain the final result. Fig.5 shows an example of color removal.

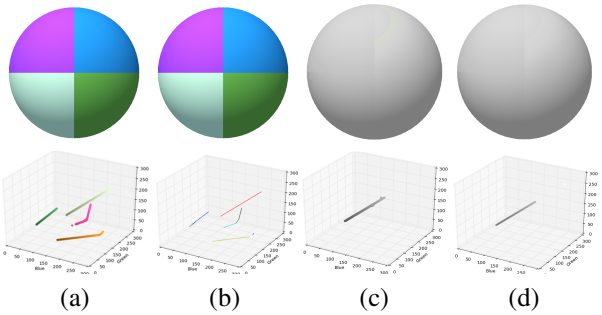

Fig. 5. An example of multicolor removal. (a) is the input image and it's RGB distribution. (b) is the generated Color Lines. (c) is the result of transform model. (d) is the final result after applying linear luminance.

4. EXPERIMENTS

We apply our method to images of objects in multicolor (Fig. 6), and compare it to Linear Luminance. As an instance, results of SFS of teapot is shown in Fig.7. Here we use 4 SFS methods, including TS by Tsai and Shah[5], FS by Falcone and Sagona[14], DD by Daniel and Durou[15] and VSY method by M. Visentini-Scarzanella et al[16].

For each input image, we take SFS results of the same objects in single color as the ground truth, and calculate the

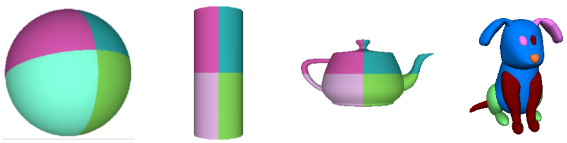

Fig. 6. Objects used for experiment

mean absolute depth error and standard deviation of absolute depth error for linear luminance and our method, as is shown in Table 1 and Table 2. It can be shown from the tables that results of SFS using output of our method get less error compared to using output of linear luminance based method.

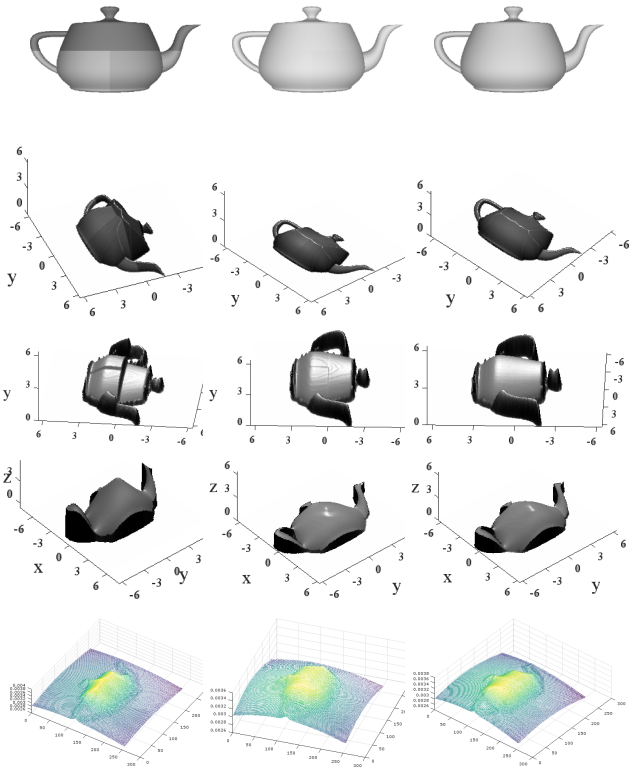

Fig. 7. Results of SFS. From top to bottom: input image, results of TS, FS, DD and VSY. From left to right: gray-scale from linear luminance, our method and ground truth.

5. CONCLUSIONS AND FUTURE WORK

In this paper we proposed a novel approach for transforming multicolor image into gray-scale, removing intensity difference between colors. This method is based on line models. To generate line models in RGB space, we proposed an improved Hough Transform approach in 3D space to classify colors, and to help generate Color Lines simply. Then we transformed the problem of scalar value adjustment into a problem of geometric transformation in RGB space. The results showed that by adopting this preprocess before SFS processing, better shape reconstruction is obtained. In the near future we are going to do some more research on the geometric transformation for the color adjustment to improve the result better, and compare the performance with other multicolor remval methods. And comparison between other multicolor removal methods will be shown in next paper.

Table 1. Mean of absolute depth errors

		Sphere	Cylinder	Teapot	Dog
TS	LL	0.33	0.11	0.07	0.15
	Ours	**0.11**	**0.06**	**0.02**	**0.06**
FS	LL	0.46	0.16	0.11	0.41
	Ours	**0.34**	**0.02**	**0.03**	**0.07**
DD	LL	2.02	1.18	0.28	3.30
	Ours	**0.31**	**0.05**	**0.06**	**0.62**
VSY	LL	6.60E-05	5.65E-05	9.55E-05	2.93E-05
	Ours	**2.51E-05**	**9.09E-06**	**8.36E-05**	**8.47E-06**

Table 2. Std of absolute depth errors

		Sphere	Cylinder	Teapot	Dog
TS	LL	0.27	0.17	0.14	0.30
	Ours	**0.08**	**0.10**	**0.04**	**0.10**
FS	LL	0.51	0.35	0.27	0.92
	Ours	**0.29**	**0.03**	**0.07**	**0.48**
DD	LL	1.41	1.82	0.53	4.28
	Ours	**0.28**	**0.08**	**0.14**	**0.93**
VSY	LL	1.01E-04	1.04E-04	2.00E-04	6.68E-05
	Ours	**1.94E-05**	**1.99E-05**	**1.33E-04**	**1.77E-05**

6. REFERENCES

[1] Gang Zeng, Yasuyuki Matsushita, Long Quan, and Heung-Yeung Shum, "Interactive shape from shading," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2005, vol. 1, pp. 343–350.

[2] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah, "Shape-from-shading: a survey," *Pattern Analysis and Machine Intelligence*, vol. 21, no. 8, pp. 690–706, 1999.

[3] Jean-Denis Durou, Maurizio Falcone, and Manuela Sagona, "Numerical methods for shape-from-shading: A

new survey with benchmarks," *Computer Vision and Image Understanding*, vol. 109, no. 1, pp. 22–43, 2008.

[4] Guohui Wang, Shangzheng Liu, Jiuqiang Han, and Xinman Zhang, "A novel shape from shading algorithm for non-lambertian surfaces," in *Measuring Technology and Mechatronics Automation*. IEEE, 2011, vol. 1, pp. 222–225.

[5] Tsai Ping-Sing and Mubarak Shah, "Shape from shading using linear approximation," *Image and Vision computing*, vol. 12, no. 8, pp. 487–498, 1994.

[6] Rui Huang and William AP Smith, "Shape-from-shading under complex natural illumination," in *International Conference on Image Processing*. IEEE, 2011, pp. 13–16.

[7] Tal Hassner and Ronen Basri, "Example based 3d reconstruction from single 2d images," in *Conference on Computer Vision and Pattern Recognition Workshop*. IEEE, 2006, pp. 15–15.

[8] Xinyu Huang, Jizhou Gao, Liang Wang, and Ruigang Yang, "Examplar-based shape from shading," in *International Conference on 3-D Digital Imaging and Modeling*. IEEE, 2007, pp. 349–356.

[9] Moumen T El-Melegy, Aly S Abdelrahim, and Aly A Farag, "Better shading for better shape recovery," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2014, pp. 2307–2312.

[10] MKM Rahman, Tommy WS Chow, and Siu-Yeung Cho, "A segmentation based approach for shape recovery from multi-color images," pp. 659–663, 2010.

[11] Xiaozheng Zhang, Yongsheng Gao, and Terry Caelli, "Colour adjustment and specular removal for non-uniform shape from shading," pp. 563–568, 2010.

[12] Ido Omer and Michael Werman, "Color lines: Image specific color representation," vol. 2, pp. II–946, 2004.

[13] John Illingworth and Josef Kittler, "A survey of the hough transform," *Computer vision, graphics, and image processing*, vol. 44, no. 1, pp. 87–116, 1988.

[14] Maurizio Falcone and Manuela Sagona, "An algorithm for the global solution of the shape-from-shading model," in *International Conference on Image Analysis and Processing*. Springer, 1997, pp. 596–603.

[15] Pascal Daniel and Jean-Denis Durou, "From deterministic to stochastic methods for shape from shading," in *Proc. 4th Asian Conf. on Comp. Vis*, 2000, pp. 1–23.

[16] Marco Visentini-Scarzanella, Danail Stoyanov, and Guang-Zhong Yang, "Metric depth recovery from monocular images using shape-from-shading and specularities," in *International Conference on Image Processing*. IEEE, 2012, pp. 25–28.