Cel Shading / Toon Shading on Glass and Metal Objects

DH2323 Computer Graphics and Interaction Project Report

Bosen Cheng

bosen@kth.se

Royal Institute of Technology KTH.

ABSTRACT

In this project, I explore a stylish non-photorealistic rendering method, named Cel Shading or Toon Shading, which makes the 3D computer graphics appear to be flat. Firstly, I will talk about the purpose of this project in the Introduction. In the Background, I will introduce the development of the technique. Then I will show one simple scene to demonstrate the basic idea of cel shading and then implement two scenarios in Blender and Unity. In the end, I will discuss future work from more perspectives of such techniques.

1 INTRODUCTION

The word "cel" in cel shading comes from the material celluloid, which was used for traditional 2D animation painting in the last century for a long time. Thus, people use this word to represent 2D animation. Traditionally, drawing 2D animation for movies or games requires a heavy workload when there is a lot of movement. With the development of computer graphics, designers nowadays can imitate 2D animation by rendering 3D models in a stylish way and reducing the cost. However, it is not easy to achieve a satisfying cartoon-like effect on glass and metal objects when rendering with such a method. In this project, I will study the basic concept of cel shading and explore different ways to implement a cel shader on glass and metal objects using Blender and Unity.

Figure 1 ***The Legend of Zelda: The Wind Waker*** **game screen shot.**

2 BACKGROUND

2.1 Video Games

The video game is the industry that first deploys cel shading techniques commercially. One of the famous examples is *The Legend of Zelda: The Wind Waker* in 2002 (shown in Figure 1) [1]. The cel-shaded models in the game provide players with a very distinct experience with such cartoon style. This type of non-photorealistic rendering gives the designer choices to equip their product with a special feature to attract players. *Borderlands* is a series of games that feature its stylish artistic design (shown in Figure 2) [2].

Figure 2 ***Borderlands 1, 2, 3*** **(from left to right) in-game CG screenshot.**

2.2 Animations and Movies

Cel shading in animations and movies is generally used to design models that are more easily done in 3D. For example, the film *The Iron Giant* merged traditional animation with computer techniques. Most of the scenes in the movie remain 2D, while the giant robot is a cel-shaded 3D model [3]. Figure 3 shows a typical scenario where cel-shaded models (the blue robot and cars) and 2D characters are combined [4].

Figure 3 ***Ghost in the Shell: Stand Alone Complex*****, episode 3.**

Also, there are animations and movies that are entirely made with the cel shading technique. *Appleseed* in 2004 is one interesting example [5]. In this movie, the human characters are all cel-shaded while the other models like the robots, buildings are kept with relatively realistic 3D effect (shown in Figure 4), which creates strange rendering results and a feeling of Fragmentation. Another example is *Land of the Lustrous* [6], where the characters' hair (and their bodies) in the original

comic are made of different gemstones. The animators of this animation use cel shading to create the shiny hair of those characters while making the overall scene consistent.

Figure 4 ***Appleseed*** **2004.**

Figure 5 ***Land of the Lustrous*****, episode 1.**

3 IMPLEMENTATIONS

3.1 Basic Concept

Generally, in the cartoon, or 2D animation, the models would have fewer color tones, and their outlines are explicitly drawn. Thus, I will implement these two features. Less color is relatively easy to achieve. Here, we have the constant indirect light, and therefore the only thing is to make the continuous direct light discrete by making the angle dot product discrete. The direct light here is calculated by

$$\boldsymbol{R} = Power \times \max(\hat{r} \cdot \hat{n}, 0).$$

Instead of directly take $(\hat{r} \cdot \hat{n})$ as the input, we divide them into few steps. For instance, if we want three color tones, we can set a step function:

$$y = \begin{cases} 0.2, & 0 < (\hat{r} \cdot \hat{n}) < 0.4 \\ 0.55, & 0.4 \leq (\hat{r} \cdot \hat{n}) < 0.7 \\ 0.8, & 0.7 \leq (\hat{r} \cdot \hat{n}) < 1 \end{cases}$$

And take y as the input when calculating $\boldsymbol{R}$. The results are shown in Figure 6 with the original shading as a comparison. After this, we need to draw the outline of the models.

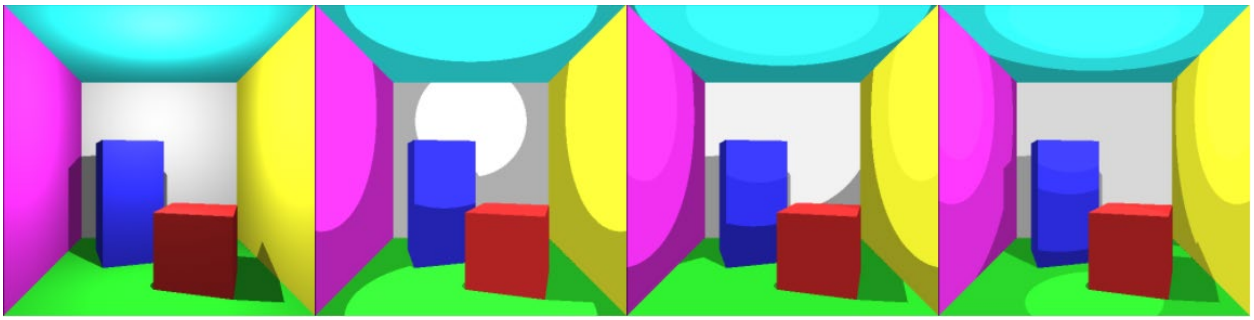

Figure 6 Original model vs. Cel Shading (2 to 4 colors tones)

There are three typical methods to draw the outline for 3D models. The first one is to calculate the face direction of all angles and draw the edges that are included both by front-face triangles and back-face triangles. The pseudo-code for this algorithm is as follows:

Step-1: for each triangle in the model:
    Calculate whether the triangle is front or back

Step-2: for each edge in the model:
    If an edge is contained in both the front triangle and the back triangle:
        Draw this edge as an outline

The core idea of the second method is to extend the model's vertices along the normal direction, then fill the enlarged model with black color (usually only fill the back face), and then re-render the original model on top of it (workflow is shown in Figure 7). This algorithm can be implemented in the shader and used in the Unity part of this project.

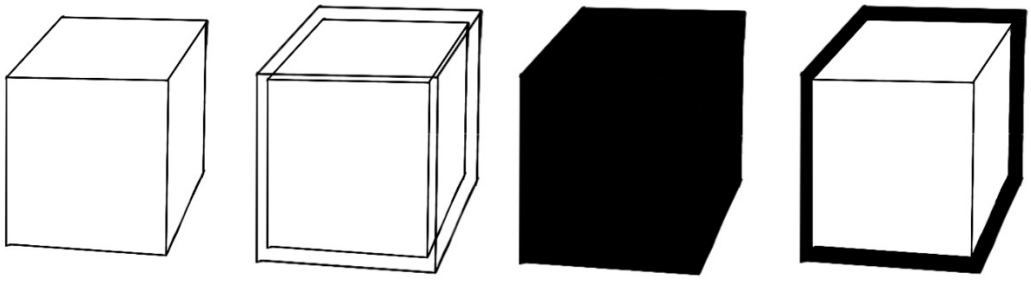

Figure 7 Cel shading outline workflow (drawn by me).

The last one is using the image processing method. The normal vector of triangles or depth can be taken as the input of the edge detection algorithm, which detects the normal vector and depth mutations in the picture and uses them as outlines. Here, I chose to use the Canny algorithm (packaged in OpenCV) to achieve edge detection here (shown in Figure 8). I deploy the Canny algorithm with depth information and grayscale color converted from models' original color. The conversion in OpenCV is calculated by [7]:

$$grayscale = 0.1140 \times \text{color.b} + 0.5870 \times \text{color.g} + 0.2989 \times \text{color.r}$$

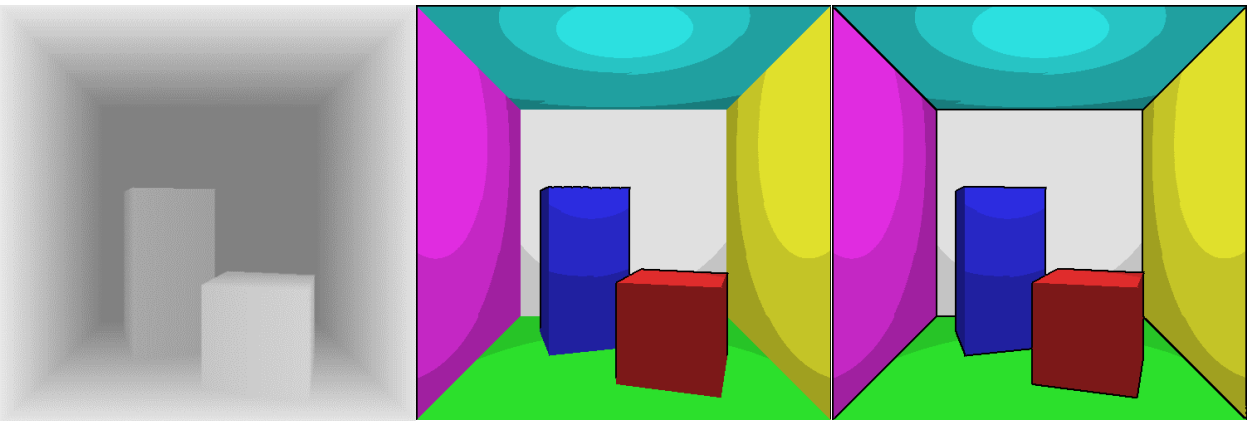

Figure 8 Depth Figure vs. Outlined Figure (with Depth Data) vs. Outlined Figure (with Grayscale color)

3.2 Cheating Way Achieving Cel Shading without Outline

Since it is easier to achieve the reflection and refraction using path tracing, I find a cheating way to create the cel-shading effect to some degree. Here, I make the dot product of the light vector and surface normal discrete, thus reducing the color tunes, which creating an oil-painting-like effect on the resulting figure (Figure 9).

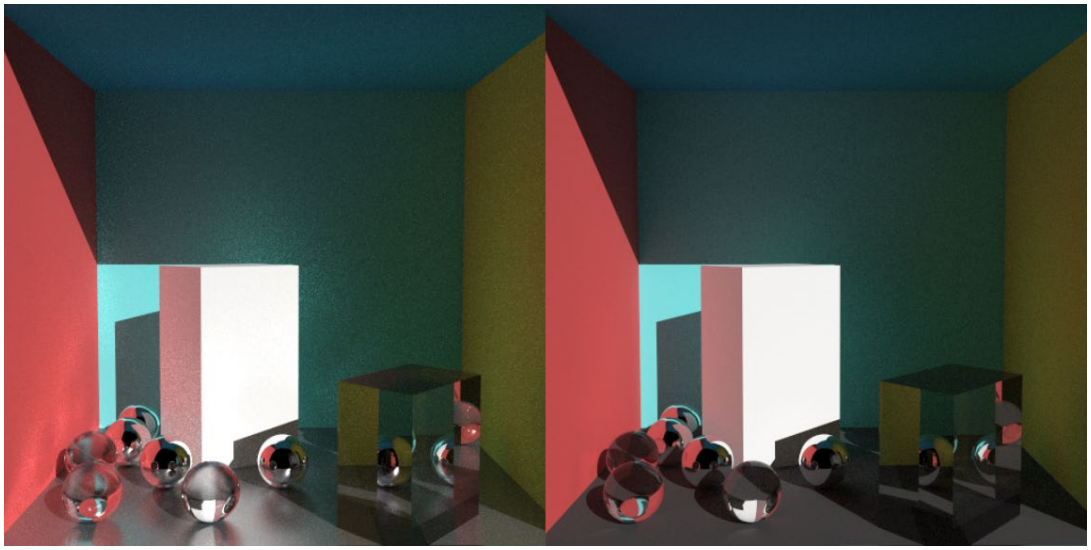

Figure 9 Cheating Way of Cel Shading using Path Tracing.

3.3 Cel Shader in Blender

In Blender, the basic process is to modify the color output from an existing shader. For the glass cup, I used the color ramp node to make the continuous color on the glass discrete and mixed this color with a transparent node to offer a transparent feeling. For the metal teapot, I used two color ramp nodes, one is to set its surface color, the other one is to create the discrete color for the highlight and shadow of the metal object. The result is shown in Figure 10.

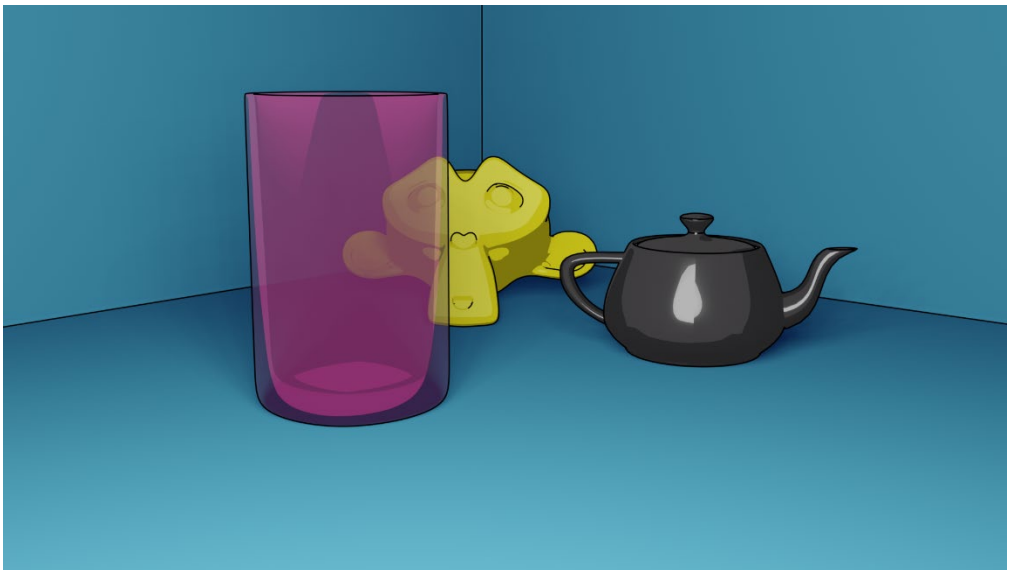

Figure 10 Cel Shaded Glass Cup and Metal Teapot in Blender

3.4 Cel Shader in Unity

For cel shader in Unity, the hardest parts are the specular light. As for the metal teapot, the cel shader calculates the specular light and highlights independently apart from the normal light. For the glass cup, the shader needs to calculate one more thing, the rim light caused by the Fresnel effect. Then I used the step function to make the color in the shader discrete. The result is shown in Figure 11.

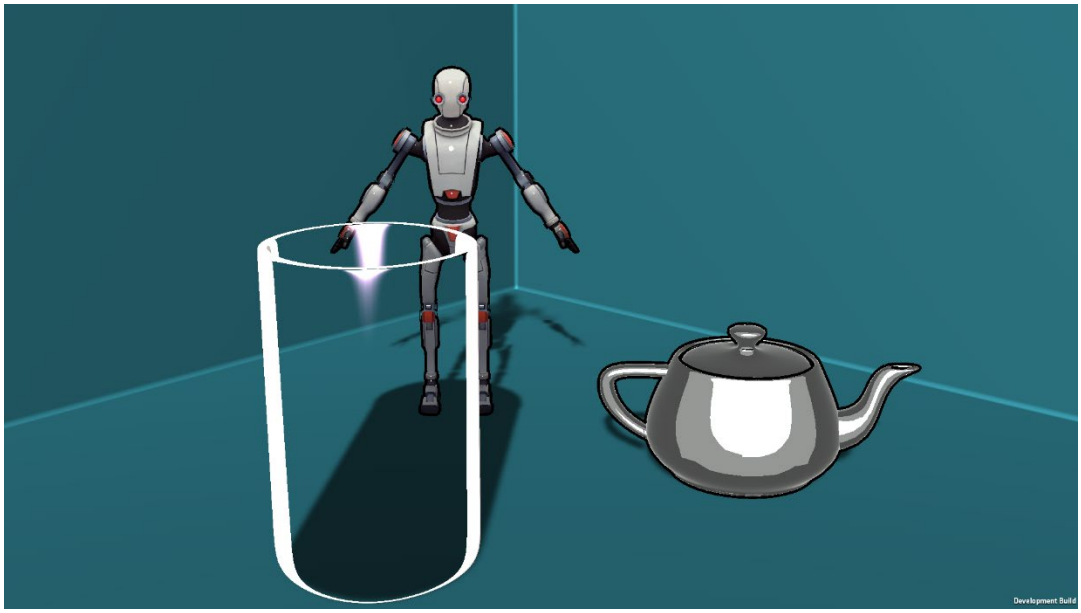

Figure 11 Cel Shaded Glass Cup and Metal Teapot in Unity

FUTURE WORK

For future work, we do not need to restrict ourselves to only shader level, rendering a 3D model in 2D style in fact, covers many aspects of a render pipeline. For example, there is more than one style of cel shading, and one can make the scene looks like drawn by watercolor pen by adjusting the alpha

channel of the color (shown in Figure 12 [8]). Besides, the texture and the material of a model also influence the render results. We can dive further into one of these aspects.

Or we could also go into the direction of perception. For example, cel shading is especially useful for video games that are based on anime or comics. As both an audience and a player, one would like the game to persist the original drawing style. The problem here is to study to what degree people are able to distinguish between the traditional 2D drawing animations and 3D video games rendering with Cel shading.

Figure 12 Opacity Brush Outline Scene [8].

REFERENCES

[1]. Nintendo EAD. (Developer) (2002) *The Legend of Zelda: The Wind Waker* [Nintendo GameCube, Videogame]. Japan, Nintendo.

[2]. Gearbox Software. (Developer) (2012) *Borderlands* series [Multiplaform, Videogame]. USA, 2K Games.

[3]. Brad Bird (Director) (1999) *The Iron Giant* [Cinema, Movie]. USA, Warner Bros.

[4]. Kenji Kamiyama (Director) (2002 - 2005) *Ghost in the Shell: Stand Alone Complex* [Animation Series]. Japan, Production I.G.

[5]. Shinji Aramaki (Director) (2004) *Appleseed* [Cinema, Movie]. Japan, Toho.

[6]. Takahiko Kyogoku (Director) (2017) *Land of the Lustrous* [Animation Series]. Japan, Orange.

[7] OpenCV (Retrieved 2021) *Color Conversions*.

https://docs.opencv.org/3.4/de/d25/imgproc_color_conversions.html

[8] Luque, Raul (2012). *The Cel Shading Technique*.