Rendering Assignment

Eimear Crotty
Bournemouth University

ABSTRACT

This report is for the rendering assignment. The code which is described in this report can be found at https://github.com/eimearc/rendering.

1 INTRODUCTION

The goal of this assignment is to model and render a simple real-world object using the Renderman API.

2 SELECTED OBJECT

I selected a clean mug for the assignment (see figure 1). This mug is approximately 11.5cm in height.

Figure 1: The mug.

3 MODELLING AND UVS

Modelling

The mug consists of two parts, the cylindrical body and the handle. When being modeled in clay, they are made separately, but attached later.

I modeled both the body and the handle completely using the Renderman Python API. The bottom of the mug is tiered, in that it doesn't form a 90 degree angle with the base. I created a Python class for describing a ring of vertices, which was then used as the basis for creating all the necessary parameters to pass in to the ri.SubdivisionMesh call. Each edge loop has its own sharpness, which helps define the edges in the necessary detail without adding too many edge loops.

In addition to the extra detail on the bottom, the top of the mug has a slight lip, which I added using extra edge loops in Python. These also have their own sharpness. While these steps could have been completed using displacement, I used edge loops as it did not add too much complexity to the model in terms of polygon count and I found I had much more control over the roundness and exact shape of the lip.

I followed the same approach for the handle. There is an added complexity to the handle in that vertices fan out slightly where the handle joins with the mug body, to give more support to the handles. I instrumented my code to handle this case.

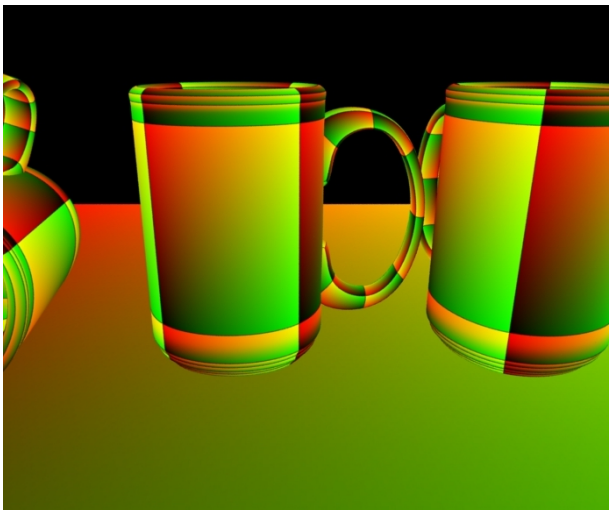
Figure 2: The lip at the top of the mug.

UVs

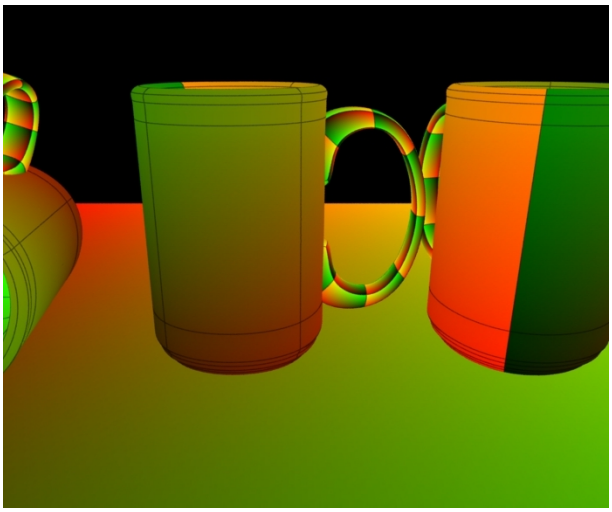
I explicitly created custom UVs (figure 3) for the body of the mug. This gave me more control over the placement of noise and displacement, which is important for creating a realistic image. This was a complex process and involved unwrapping the body into one UV shell. However, this extra effort made placing the logo and controlling the direction of some of the scratch marks incredibly easy.

4 BRDF

A ceramic mug is finished by firing the model, painting it and finally glazing it. This results in two distinct layers - an underlying colour (white in this case) and a thin layer of glaze. The paint layer is mostly uniform, with little variance of colour over the model. The glaze layer has more variance in that it is not perfectly specular over the entire surface.



(a) Default UVs.



(b) Custom UVs.

Figure 3: UV generation.

Figure 4: The model.

Figure 5: The handle.

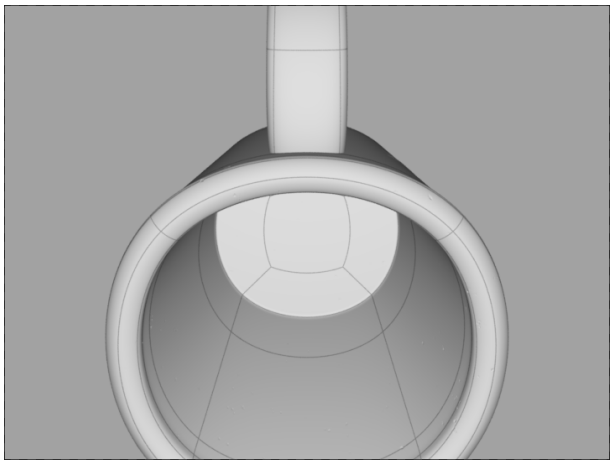


Figure 6: The inside of the mug.

This is due to people handling it, or from the glaze wearing down from wear and tear.

I used the PxrSurface BxDF. In addition to the diffuseColor parameter (which I will describe in 5), I added a clearcoat to simulate the glaze. I fed in a shader output to the clearcoatRoughness to drive the variance in specularity over the surface.

5 SURFACE DETAIL

The mug I referenced is a few years old and has accumulated damage from use. There are a variety of details that I found on the mug.

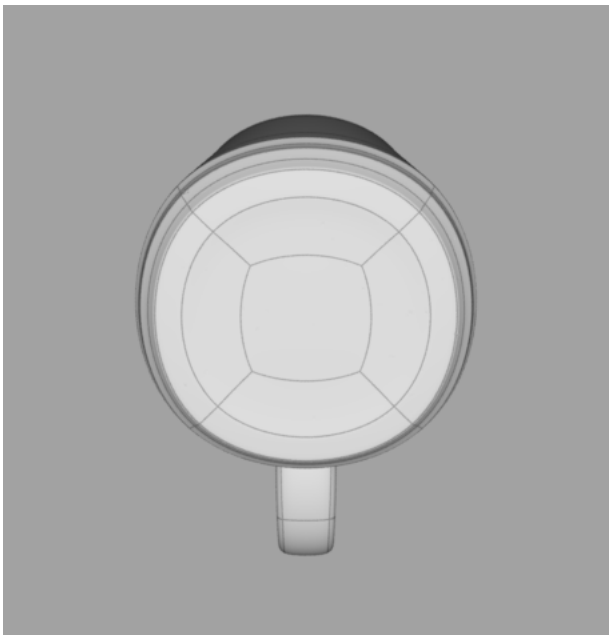


Figure 7: The bottom of the mug.

Figure 8: Small spots on the mug.

Figure 9: Smudge mask.

Scratches

These are surface details that don't have any depth. They are created, most likely, from metal spoons, resulting in a thin grey line on the surface. These scratches are generally straight and thin to the point of being unnoticeable.

There are two sets of straight scratches generated in *scratch.osl*. I then use smoothstep to control the amount of scratches. The generated mask is used in combination with the input colour to output the scratches with the correct grey colour.

Spots

These are different from the dents as, like the scratches, they result only in a change of colour on the surface. These spots are grey in colour and cover very little surface area of the mug.

I used multiple instances of perlin noise to create the spots in the shader. I used smoothstep to ensure there were no harsh edges between the spots and rest of the surface.

Dents

There are also very small, hardly noticeable dents from either the manufacturing process or from use. There are not many of these covering the mug.

I used the output of a PxrVoronoise pattern as the base for the noise. I then piped this through a smoothstep function with a very low minimum. This resulted in a small number of tiny dents in the mug, which is fed into the PxrDisplacement node.

Smudges

Smudges result from handling and change how shiny the surface looks. They only influence the specularity of the surface, not the colour.

Using a similar technique to before, I generated many instances of perlin noise and overlayed them. I used the magnitude of this to control the clearcoatRoughness parameter of the BxDF. This results in a subtle change in specularity across the surface of the mug.

Logo

The logo shows different signs of wear and tear. The logo is slightly raised from the surface. There are areas of the logo which have chipped off completely, showing the white of the layer underneath. The remaining areas of the logo are either fully raised or half raised, resulting in a mottled look. See figure 10 for the generated displacement map.

Figure 10: Displacement map with mottled effect for the logo.

Figure 11: The logo.

I used a shader (*logo.osl*) to get the correct appearance and combination of the logo displacement and damage. The texture is placed using *st* coordinates (which I created in the modeling stage). The chips are generated using a PxrVoronoise pattern and passed into the shader, using smoothstep to control the amount of chipping. This mask is used to flatten the displacement in areas where the chips are. Another level of chips is generated and used to control areas where the logo paint remains, but the level of displacement is slightly lower than the logo. These two displacements are combined and used in an ri.Displace node. To get a smooth displacement, I set the micropolygon length. The logo texture was downloaded from Google (2020).

```
ri.Attribute(
    "dice",
    {"float micropolygonlength":1}
)
```

Figure 12: Displacement.

6 RENDER

Lighting

I used a HDRI which matches my indoor setup to light the scene. The HDRI was downloaded from Zaal (2019). I rotated it to get a nice sheen on the object.

Background

For the background of the scene, I created a simple plane for the wall and a set of repeating planes for the table. I downloaded a simple wood texture from Tuytel (2018) to add detail to the table. The wall uses a flat colour.

Denoising

I used a shading rate of 0.1 and a pixel variance of 0.01 to render the scene at 1080p. However, this was not enough to remove all of the noise in the scene. I set up all of the required AOVs and then fed the result through to the Renderman denoiser. This resulted in a sufficiently noise-free image.

Depth of Field

Photos of small objects such as the mug generally have a shallow depth of field. I replicated this using the ri.DepthOfField option. This helps focus the eye towards the subject of the image and soften the image overall.

REFERENCES

Google, , 2020. *Law and Order Logo* [online]. Place of publication: USA. Available from: https://images.app.goo.gl/6ah54Skjr9KDdy8H6 [Accessed 1 November 2019].

Tuytel, R., 2018. *Texture haven - old wood planks* [online]. Place of publication: USA. Available from: https://texturehaven.com/tex/?c=wood&t=old_planks_02 [Accessed 1 November 2019].

Zaal, G., 2019. *HDRI haven - lebombo* [online]. Place of publication: USA. Available from: https://hdrihaven.com/hdri/?h=lebombo [Accessed 1 November 2019].