A Photo Realistic RenderMan Grapefruit Shader

Nikitha Ann George*
MSc Computer Animation and Visual Effects
Bournemouth University 2015 - 2016

Abstract

This report details the development process for creating a RenderMan shader for the Rendering Project at Bournemouth University 2015 -2016 . The papers starts with a brief *Introduction* which then looks into the *Primitive Modeling* with a section on *Shaders*. Finally, the *Camera Artefacts* as well as other *Scene setup* features have been discussed.

Keywords: Renderman, Grapefruit, RIB, Shading Language, RSL

1 Introduction

(a) *A natural grapefruit [Holcomb]*

(b) *Color variation [Davis]*

(c) *Texture variation and Specular Components [Hill]*

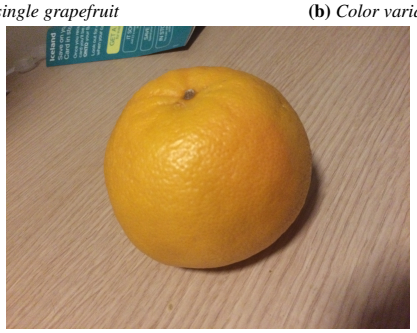
Figure 1: *Grapefruit Images taken from Internet*

The project involved getting accustomed to using RenderMan and its associated API to model and render a simple scene containing an object. A Grapefruit was selected as my object due to the simplicity in its shape, making it suitable to easily recreate with RIB primitives and having shaders than can produce a similar skin texture. **PRMan** has been used for rendering and *Image Tricks* (it) has be used to view the rendered images.

*iNumber : i7447340

(a) *A single grapefruit*

(b) *Color variation*



(c) *A second view*

Figure 2: *Grapefruit Images taken from Camera*

2 RIB Modelling

A grapefruit can be seen as a simple sphere with some elongation in its scale. A sphere is used with some scaling in the *x* and *z axis*. In addition to the existing textures on a grapefruit, a hole is seen on the top. This hole from which the fruit has been plucked is recreated within the displacement shader as well.

3 Shaders

Two shaders have been used for the Grapefruit - a *surface shader* and a *displacement shader*. An oak shader has been used for the ground plane which is found from the RenderManPro Server example files [1].

The surface shader was first written in *.sl* for testing and then later converted to *.osl*. This *osl* file has been compiled using the command `oslc` . The *.oso* is passed to the `Pattern "PxrOSL"` for appropriate use.

A displacement shader is written in *.sl* compiled with the command `shader` to produce a *.slo*. This *.slo* file has been incorporate onto the model using `Displacement` for the required texture.

3.1 Color

On close inspection with the fruit, it can be seen that the `specular` component is mid way while the `roughness` is less.

[1] RenderManProServer-20.8/lib/examples/RIS/scenes/bxdf/pattern/osl/shader/oak.osl

The surface shader for the grapefruit was developed in as stages which is described in detail as follows.

3.1.1 Surface Ramp

A grapefruit has a similar color to that of an orange. By

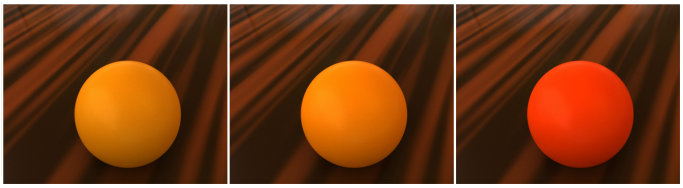
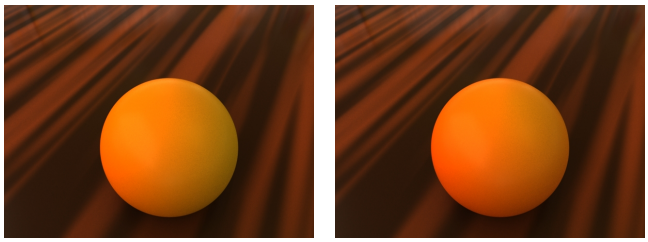


Figure 3: *The three colors used within code with lighting. From left to right :* ***lightyellow darkorange darkorange2***

looking at a natural grapefruit, a base color was formed by using a simple *s* ramp to create a gentle shading. `Ct =`



(a) *s ramp* **(b)** *s and t ramp*

Figure 4: *Results achieved with ramps*

`mix(darkorange,lightorange, ss1);`[2]
with `ss1` being the *x* position of the point in *shader space*.
A ramp based on *y* has also be introduced into the shader.

`Ct = mix(Ct,darkorange2, t);`[3]

3.1.2 Noise

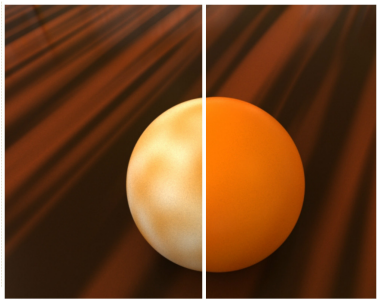


Figure 5: *Noise*

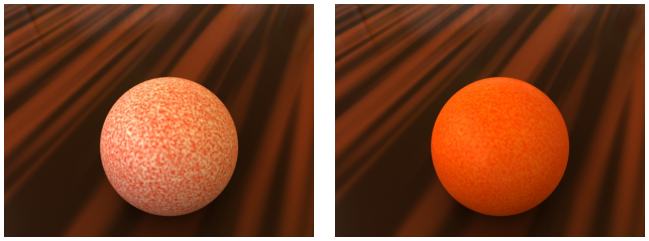
A overall noise like pattern is seen on the skin surface. This has been controlled by adding a `noisecontrol` which is a flag value to either add or remove this noise.

```
mag = mix(0,0.9,(float) noise(PP*2));
Ct = mix(Ct,darkorange, mag);
```
[4]

[2] GSurface_v02.osl: `Line 35`
[3] GSurface_v02.osl: `Line 36`
[4] GSurface_v02.osl: `Line 44`

3.1.3 Dots

On examining the surface with more detail, we can see many dots that cover the whole surface. We use the following to achieve this,
`float dots = (float) noise(PP*20);`[5]



(a) *Image depicting dots* **(b)** *Dots on Surface*

Figure 6: *Dots*

and used `dots` to drive in the `dotcolor`.

On adding these dots made the ramp in *t* seem to be very subtle and hence this operation is done after introducing the dots.

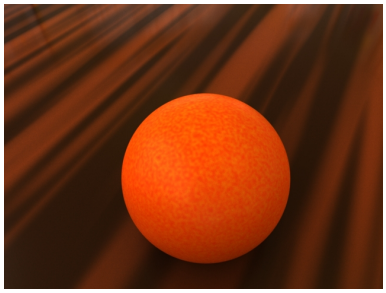


Figure 7: *Final dots implemented in shader*

3.1.4 Stem

Yellow color around stem hole

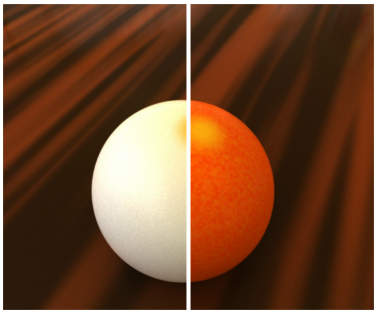


Figure 8: *Stem color - darkgold*

To create the blemish stem color, a `stemmag` and `position` has been incorporated into the network as,

```
position = smoothstep(0.01,0.04,t);
stemmag = mix(0,1,position);
```

and this is used to create the yellow color around the stem hole with,
`Ct = mix(darkgold, Ct,stemmag);`[6]

[5] GSurface_v02.osl: `Line 50 - 53`
[6] GSurface_v02.osl: `Line 58 - 69`

Stem Hole

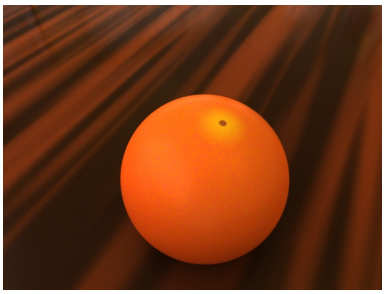


Figure 9: *Color in Stem Hole*

The stem hole itself is created by mixing brown and light-brown continuously by varying the values in the *smoothstep* for `position`,

```
position = smoothstep(0.008,0.0145,t);
stemmag1 = mix(0,1,position);
```

and plug them with the following values,

```
Ct = mix(tanbrown, Ct,stemmag1);
Ct = mix(brown, Ct,stemmag1);
```[7]

3.1.5 Spokes

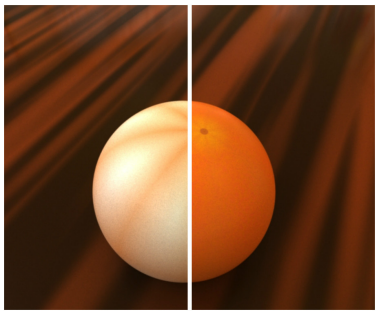


Figure 10: *Spokes*

The same spokes function described in Section 3.2.6 without `inDisk` has been used to drive the `redlines` to give the resulting image shown in Figure 10.

3.2 Displacement

A grapefruit has a very naturalistic skin texture which has been bought about in the Displacement Shader as follows. There is a couple of different varaibles and magnitudes for each of the displacement values I have used within.

3.2.1 Skin Bump

`skinmag` is a *controller variable* that denotes the area for skin-like displacement. This is a function that is a basic noise function with layering.

```
skinmag +=abs(float noise(PP*freq) - 0.5) *
2 / freq * freq;
freq *= 3;
```[8]

[7] GSurface_v02.osl: `Line 69 - 87`

[8] GDisplace_v07.sl: `Line 10 - 26`

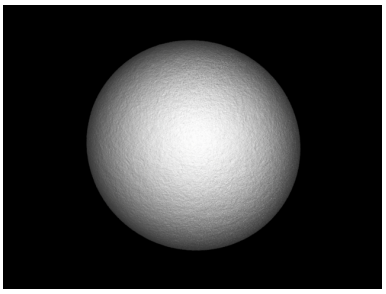


Figure 11: *Texture Skin Surface*

3.2.2 Pore Surface

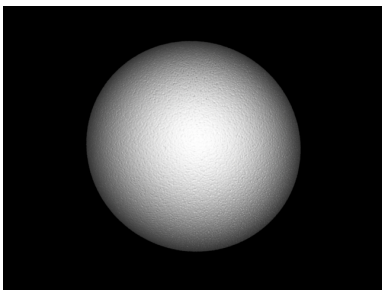


Figure 12: *Surface with Pores*

On close inspection, we are able to see tiny pores on the surface which has been created using `poremag`. I have been able to bring this in my shader using a *noise* function as well as a combinaiton of *mix* and *smoothstep* functions to drive in the holes done using `inSpot`.

```
mag = mix(0,0.4,float noise(PP*85));
inSpot = smoothstep(0.3-fuzz,0.7+fuzz,mag);
disp = smoothstep(0,1.25,inSpot);
```[9]

3.2.3 Deformation

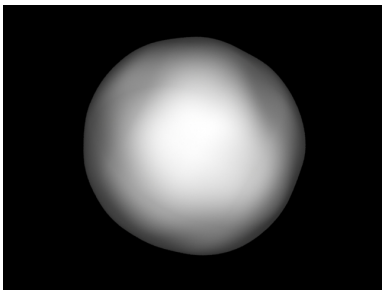


Figure 13: *The value for deformation has been exaggerated on purpose to show the effect*

A grapefruit does not have a perfect spherical shape. To bring about a bulky sphere, I have used `noisedeform` and a displacement controller `Ko`, plugged to a *noise* function with another *mix* and *smoothstep*.

```
noisedeform = smoothstep(0,0.9,float
noise(PP*2));
```

[9] GDisplace_v07.sl: `Line 31 - 43`

```
noisedeform = mix(0, 0.5,noisedeform);
```
[10]

3.2.4 Turbulence

This is a turbulence function that has been takes from the Listings in [Stephenson 2007]. [11]

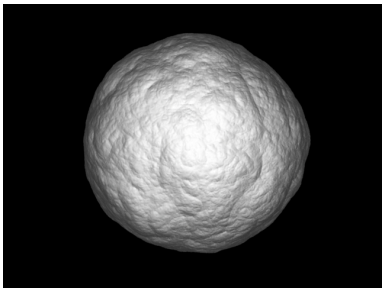


Figure 14: *Turbulence Deformation*

3.2.5 Stem

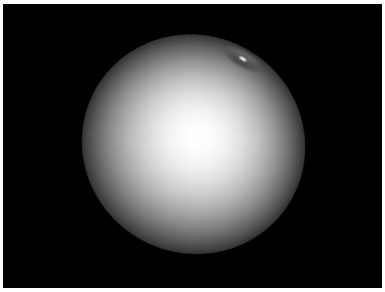


Figure 15: *The Stem*

To create the hole from the stem was one of the difficult parts for me. The first method I chose was to use the Listing in [Stephenson 2007] to first add a layer of circles, pick only a certain circle which is between t and s. But the output of this did not look very impressive and I chose to the solution I have used in my final model. I split this into two parts, a dent outside of the hole, and another set of codes for the main hole.

Stem Dent

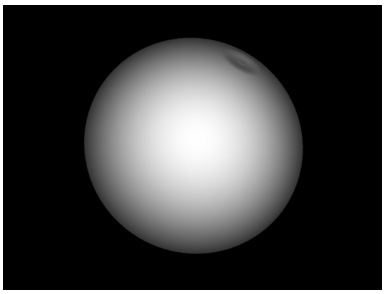


Figure 16: *An area dent around stem hole*

I have used the texture coordinate *t* to drive the *smoothstep* with values that I have written by *trial-and-error*.

[10] GDisplace_v07.sl: `Line 46 - 53`
[11] GDisplace_v07.sl: `Line 59 - 70`

```
position = smoothstep(0.01,0.06,t); stemmag
= mix(0,1,position);
```
[12]

Stem Hole

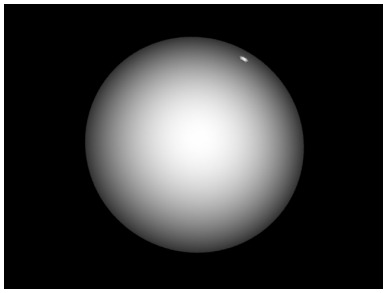


Figure 17: *Hole showing stem*

The hole was created using a similar function but I have changed the *start* and *end*.

```
position = smoothstep(0.008,0.0145,t);
float stemmag_hole = mix(0,1,position);
```
[13]

3.2.6 Spokes

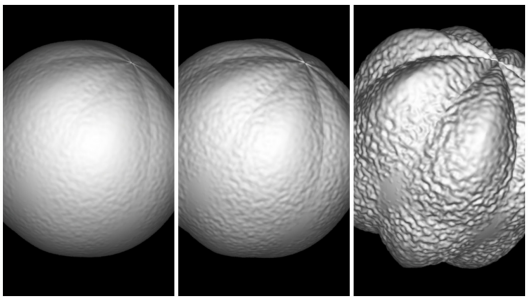


Figure 18: *Spokes with different values for displacement attenuation, spokes*

As with the hole, creating the spokes was also something I faced a lot of trouble to achieve.

The texture coordinates were modified to `ss1` and `tt1` with `ss1` multiplied by the number of spokes and `t` for some vertical alterations.

```
inDisk= 1- smoothstep(-0.1,0.55 + 0.2 *
(noise(PP * 25)),dist);
```
[14]

`inDisk` is used to cut the shape circularly. A `spokes` variable is used to check if it is within the spokes area, *i.e,* using `inDisk`. A *noise* function is used within the `inDisk` to put in a skin-like texture.

The final Grapefruit with all the displacements have been combined into the variable `mag` and have finally changed the positions and normals accordingly.

```
P = P + (mag)*Km*NN;
N = calculatenormal(P);
```
[15]

[12] GDisplace_v07: `Line 76- 85`
[13] GDisplace_v07: `Line 88 - 91`
[14] GDisplace_v07: `Line 94 - 113`
[15] GDisplace_v07: `Line 122 - 124`

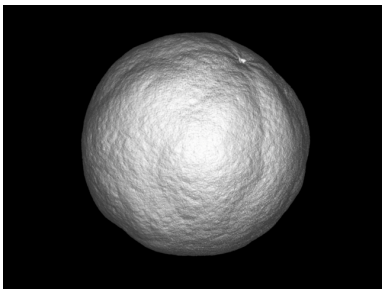


Figure 19: *Grapefruit with final displacement Shader*

where `Km` is a global displacement control and `NN` is the normalized `N`.

4 Lighting

This scene uses Geometric Area Lights using shapes *envsphere*. Though the Grapefruit does not have much reflective properties, the scene looked more realistic with the Environment Lighting.

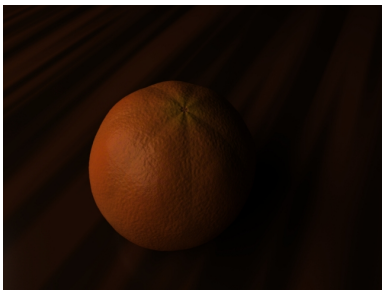


Figure 20: *Scene without Environment Lighting*

```
AreaLightSource "PxrStdEnvMapLight"
"theEnvLight" "float exposure" [0.5]
"string rman__EnvMap" ["HDRI.tx"] "float[2]
rman__ProceduralResolution" [1024 512]
Rotate -90 1 0 0
Geometry "envsphere" "constant float radius"
[30]
```
[16]

The `"HDRI.tx"` is an HDRI image incorporated into the scene, of which a more description is in Section 6. The *rectlight* is

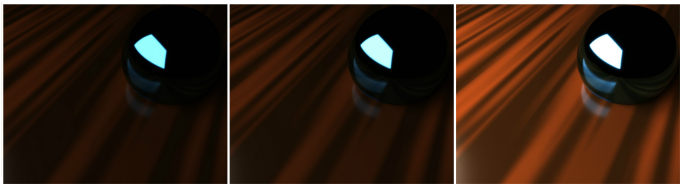


Figure 21: *Testing `rectlight` using reflective sphere with a `"rectangle Shape"` tweaking with* ***intensity****.*
From Left to Right: Intensity as ***3, 4*** *and* ***10***

taken from one of the many Pixar example codes[17].

[16] Grapefruit_v02.rib: `Line 61 - 68`
[17] Pixar/RenderManProServer-20.8/lib/examples/RIS/scenes/bxdf

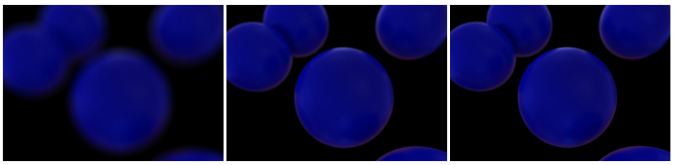
```
AreaLightSource "PxrAreaLight" "myRect"
"string shape" ["rect"] "float intensity"
[2]
Geometry "rectlight"
```
[18]

Figure 22: *Development to final lighting for scene with both lights*

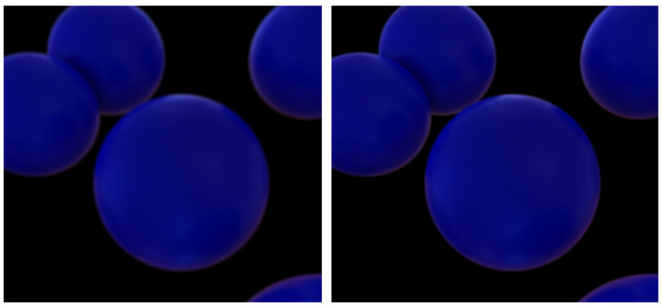
.

5 Camera Artefact

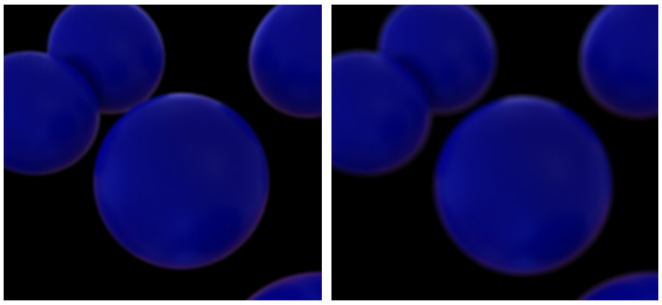
Since I have chosen an object that does not undergo a motion activity, I have decided to incorporate a *Depth of field*. The camera I have used is not equipped with such capabilities and so I improved the scene with values that looks right to my eyes. I have used some tests where the ball has been removed of color and displacement to speed up render times.



(a) *Left to Right: Focal distances of* ***2, 7*** *and* ***8***



(b) *Left to Right:f-stop of* ***2*** *and* ***3***



(c) *Left to Right:Focal Length of* ***150*** *and* ***400***

Figure 23: *Camera tricks*

`DepthOfField f-stop focallength`

[18] Grapefruit_v02.rib: `Line 71 -84`

focaldistance with
f-stop as **2**, *focallength* as **0.400** and *focaldistance* as **8**.

6 Environment Map

Figure 24: *Environment Map [Maps]*

I have used a simple HDRI environment map that is of a small apartment room I have taken from [Maps] which is in.jpg image format. I have converted this to a RenderMan texture format using,

```
txmake Room_HDRI.jpg HDRI.tx
```
and then incorporated it into the scene rib file using,

```
AreaLightSource "PxrStdEnvMapLight"
"theEnvLight" "float exposure" [0.5]
"string rman__EnvMap" ["HDRI.tx"] "float[2]
rman__ProceduralResolution" [1024 512]
```
with an environment sphere,

```
Geometry "envsphere" "constant float radius"
[30]
```

7 Further Detailing

Figure 25: *Oak Table [Maps]*

A textured wooden surface was first considered to be used as the table texture but the results look really depressing and I chose to use the example codes available. The oak shader was considered to be the right choice for the table on which the grapefruits rest. This scene setup seemed good to me and so I stopped here. The table is a patch of which the dimensions I have taken from Pixar's RenderMan examples [19]. The table has been given a texture similar to the texture of the ground seen in Figure 24 and hence a glossy look has been created. Each *Grapefruit* has been scaled and rotated accordingly to produce a visually attractive image.

[19] RenderManProServer-20.8/lib/examples/RIS/tepots/teapots_RISShaders.rib

8 Scene Setup

The following has been specified for the scene,

```
Display "Grapefruit.tiff" "it" "rgb"
Format 1280, 720 1
PixelFilter "gaussian" 2 2
Hider "raytrace" "int incremental" [1]
Hider "raytrace" "int incremental" [1]
"string integrationmode" ["path"] "int
maxsamples" [256]
PixelVariance 0.005
Projection "perspective" "fov" 50
```

with integrators,

```
PxrDefault, PxrVCM, PxrDirectLighting,
PxrPathTracer
```

In addition to the above, the scene rendered has been setup with a main Grapefruit in the middle with two other fruits rotated and scaled on its left and lastly, two similar grapefruits on the right to add to the scene. The grapefruits in the middle has a `Ko` of **0.5**. The one on the left that is futher away from the camera has **9** spokes and the `noisecontrol` is **off [0]**.

Acknowledgments

For the completion of my assignment, I thank my fellow classmates as well as the demonstrator, Mr. Constantinos Glynos at Bournemouth University for helping me setup `PRMan` on my Personal Computer. I also thank my course lecturer, *Ian Stephenson* throughout the development of this assignment. Lastly, I also thank my friends and family for giving me an opinion each time I did something new with any of the shaders.

Conclusion

Though the scene looks fairly decent, a lot more could be developed further, especially the background setup. A few more tweaking could have been done for the Lighting. Though I have tried my best to give the natural look of an imperfect sphere, a bit more could be done to develop the shape to a less perfect sphere. A more naturalistic look for the color of the stem hole could have been used. However, I had fun and learned a lot developing a shader for this assignment.

References

APODACA, A. A., GRITZ, L., AND BARZEL, R. 2000. *Advanced RenderMan: Creating CGI for motion pictures*. Morgan Kaufmann.

CHOU, K. Making an orange shader, shader exercises. Available at `http://equinoxin.blogspot.co.uk/2011/03/making-orange-shader-shader-exercises.html?m\unhbox\voidb@x\bgroup\let\unhbox\voidb@x\setbox\@tempboxa\hbox{1\global\mathchardef\accent@spacefactor\spacefactor}\accent221\egroup\spacefactor\accent@spacefactor`.

CORTES, R. 2007. *The RenderMan Shading Language Guide*. Cengage Learning.

DAVIS, P. . How to send texas grapefruit. Available at `http://www.pittmandavis.com/articles/citrus/grapefruit/send-grapefruit/how-to-send-texas-grapefruit.html`.

GALLAGHER, C. Renderman - rib wrangling. Available at https://sdm.scad.edu/faculty/mkesson/vsfx319/wip/best_fall2015/christian_gallagher/kettle/index.html.

HILL, V. Ezekkel tisztitsuk meg a mjunkat! Available at http://webshop.hillvital.hu/tisztitsd-a-majad.

HOLCOMB, M. Your pregnancy: Week 24. Available at http://www.babypost.com/pregnancy/pregnancy-guide/your-pregnancy-week-24.

MAPS, S. H. Hdrmaps. Available at http://hdrmaps.com/freebies/small-apartment.

PIXAR. Renderman 20. Available at https://renderman.pixar.com/resources/current/RenderMan/home.html.

STEPHENSON, I. 2007. *Essential RenderMan*®. Springer Science & Business Media.

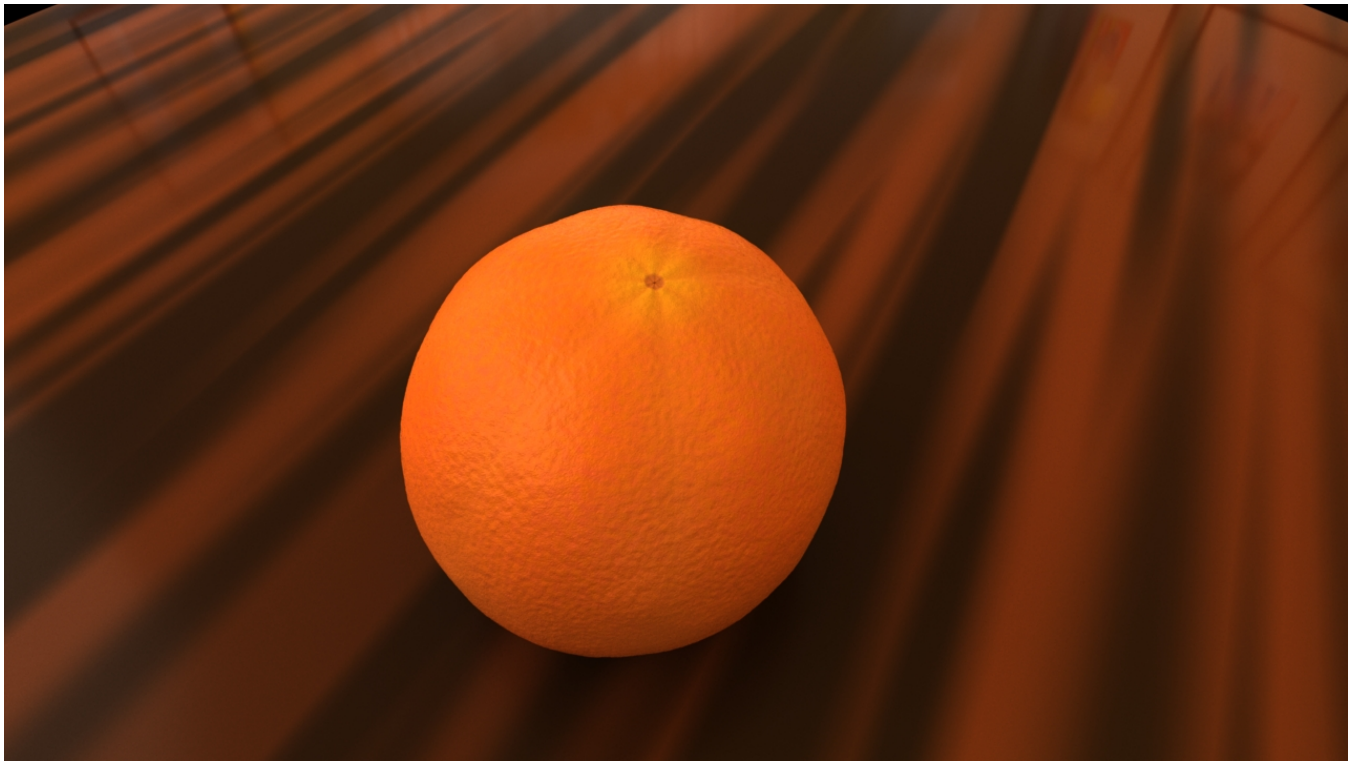


Figure 26: *Single Grapefruit*

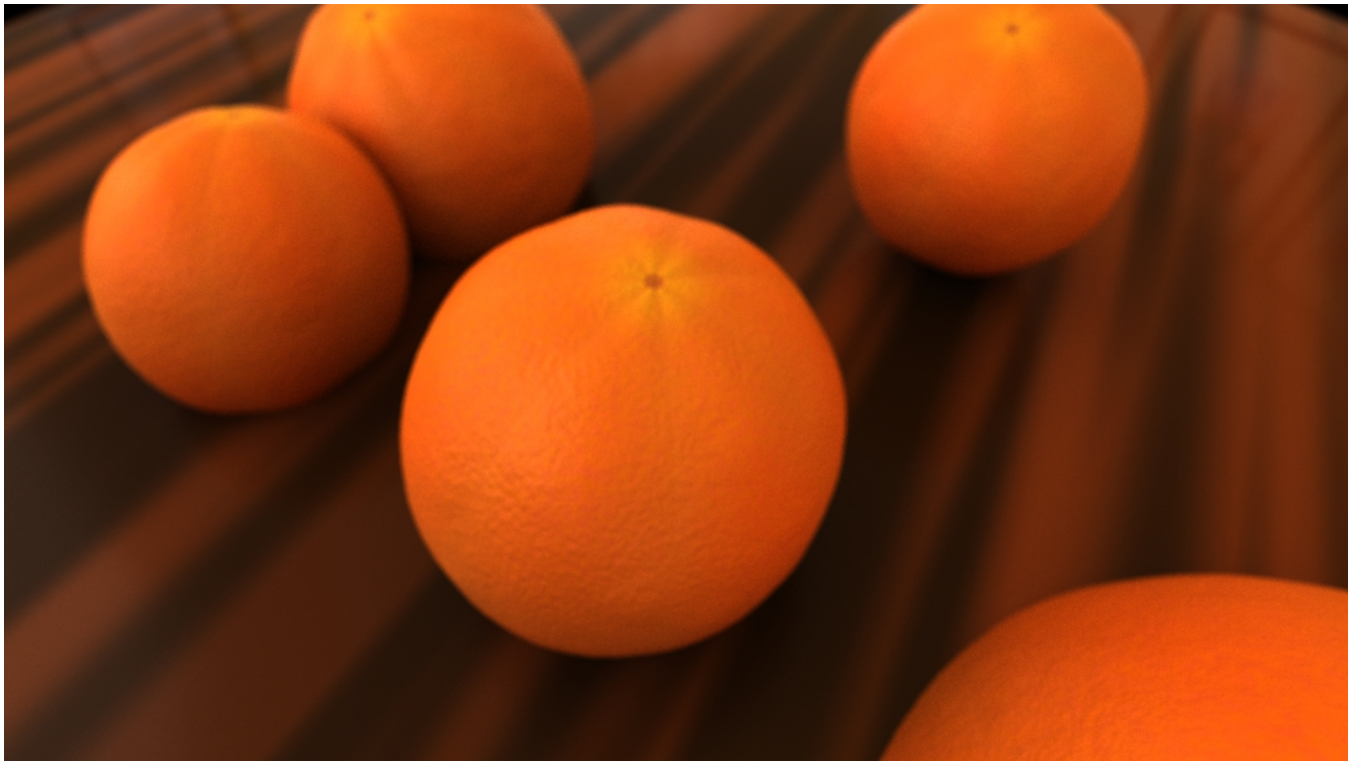


Figure 27: *Final Render with Depth of Field*

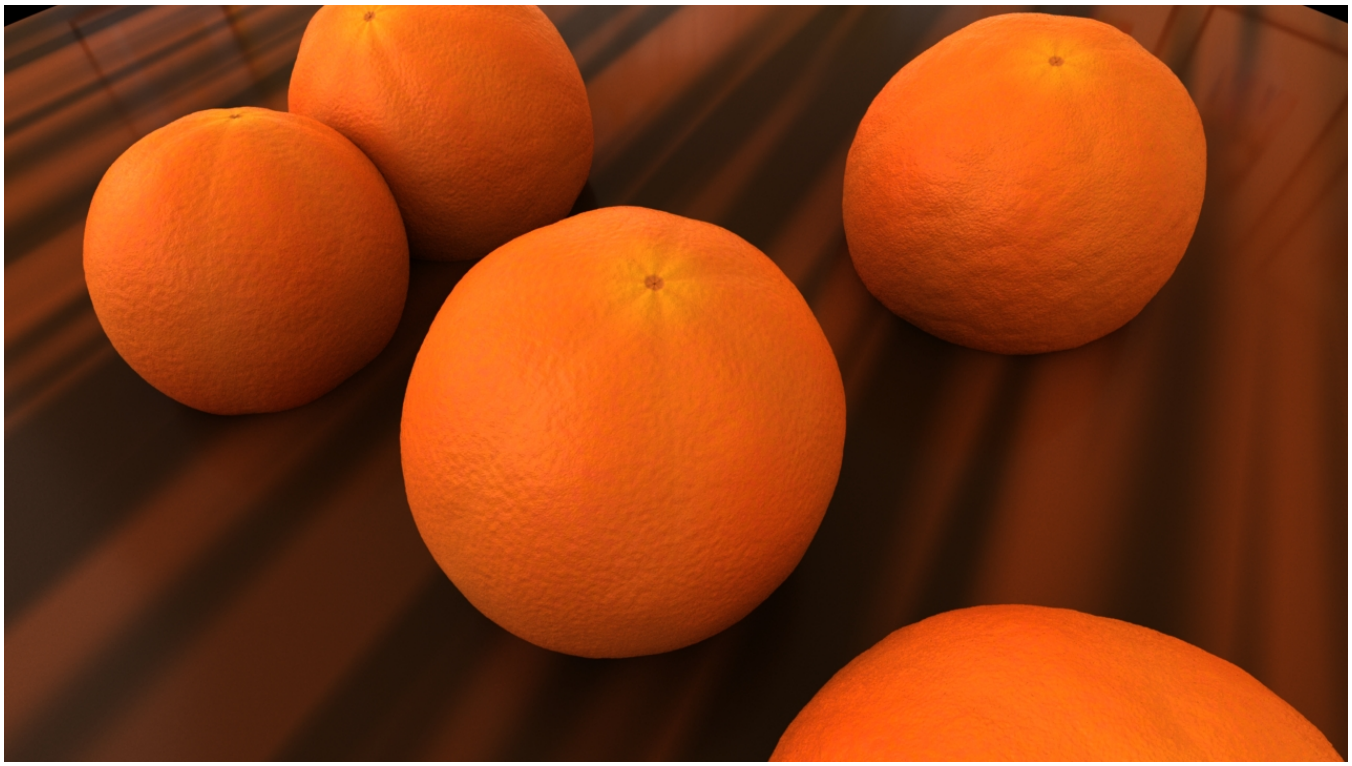


Figure 28: *Final Render without Depth of Field*