Maris Stijn

Dynamic Day and Night Cycle in a Binary Star System

Graduation work 2018-19

Digital Arts and Entertainment

Howest.be

CONTENTS

Contents 1
Abstract 3
Introduction 4
Research 4
1. Atmospheric Scatering 4
1.1. RayLeigh Scattering 4
1.2. MIE Scattering 4
1.3. NON-SELECTIVE scattering 5
1.4. analytical models 5
1.4.1. Isotropic Model 5
1.4.2. perez et al. model 5
1.4.3. Preetham et al. model 5
1.4.4. Hosek and Wikie model 6
1.5. PreComputed Models 6
1.5.1. Nishita et al. model 6
1.5.2. Bruneton et al. model 7
1.6. Spectral Models 7
2. Effects of a binary star system on planet 7
2.1. Binary star system 7
2.2. Effect on light and shadows 7
2.3. effect on the Atmosphere 7
2.4. Eclispses of one star by the other star 8
2.5. effect on the Habitibility 8
3. Effects Of different star types 8
3.1. different star types 8
3.2. effect on Light And Shadow 8
3.3. effect on the Atmosphere 8
3.4. Effect on Habitibility 9
case study 9
1. introduction 9
2. Implementing the Hosek et al. model With BluePrints 9
2.1. Creating the Shader 10
2.2. Calculating the Parameters used in the shader 10

2.2.1 Use of Data Tables 10
2.2.2 Evaluation of The Splines and calculating the Parameters 11
2.2.3 Encountered Issues 11
2.3 The result 12
3. ImplemeNting The Hosek et al. Model With C++ 12
3.1. Calculating the Parameters used in the shader 12
3.2 The result 13
4. Fixing The Graph Shader 13
4.1 Fixing The HLSL Shader 14
5. Binary star system in unreal 16
5.1 Adding Second sun to already existing Unreal Model 16
Conclusion 17
References 18

ABSTRACT

The aim of this paper is to research and implement a dynamic day and night cycle for a binary star system with different star types for a livable planet. Researching the effects and possible sky models to use for implementation. To finally have a real time simulation of the binary star system.

INTRODUCTION

A dynamic day and night cycle is used to recreate our sun, moon and sky. Accurate models are more commonly used simulators because accuracy is important. However, more accuracy usually means less performance. So, it is important to find the right balance between accuracy and performance that is right for your project. Therefore, in games where performance is very important they usually use less accurate models because as stated before they have better performance. Since the first model was described, computers and their performance have come a long way. Therefore, now there are many methods to create a dynamic day night cycle. Which is why many different options were explored and compared to find the best fitting methods to be used. Because the implementation is for a binary star system and the star type should be adjustable. Exploration was done to see if any existing model at all, could be used or adapted. Then secondary factors were important, comparing the performance and accuracy balance, what are the weak points, flexibility and adaptability in the model.

RESEARCH

A dynamic day and night cycle is a way to represent the passing of day and night in games. This is done by having a moveable sun and moon and lights and shadows that follow the sun position. A big part of this is the way the sun interacts with the atmosphere. Most of this is caused by atmospheric scattering.

In this case a look at how a binary star system would affect these natural phenomena was desired. Also, how different star types would affect them.

1. ATMOSPHERIC SCATERING

Atmospheric scattering consists of multiple factors that influence the sunlight one sees on earth. The atmospheric scattering of light occurs when light particles collide with other particles in our atmosphere. There is Rayleigh scattering, Mie Scattering, non-selective scattering.

1.1. RAYLEIGH SCATTERING

This is the scattering that occurs when light hits molecules of air about one tenth the size of the lights wavelength and is scattered in all directions this can be seen in. This scattering is wat makes our sky blue. Because Rayleigh scattering is more effective at short wavelengths.

1.2. MIE SCATTERING

Mie scattering occurs when the light hits particles similar in size than the wavelengths of light. This scattering creates the glare around the sun and the light one sees when there is fog or haze caused by pollution. Mie scattering has more of a direct effect cause the scattering is very directional. While Rayleigh scattering is indirectly more prominent. This can be seen in *Figure 1*.

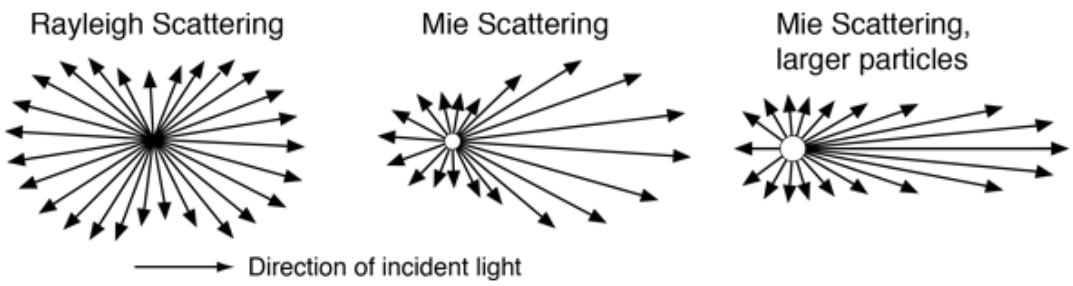

Figure 1: Representation of Rayleigh vs Mie Scattering.

1.3. NON-SELECTIVE SCATTERING

This occurs when the light hits particles larger than the size of the wavelength of light. All light is scattered evenly. A Great example of this are clouds they scatter all light evenly hence they are white.

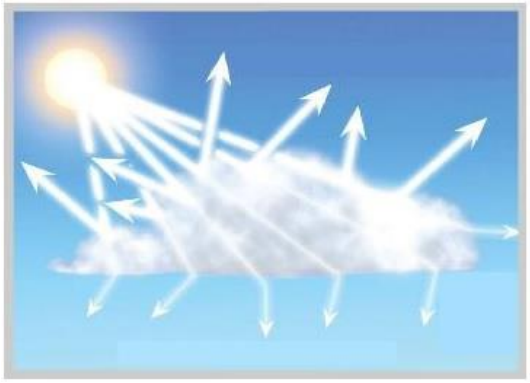

Figure 2: Representation of non-selective Scattering.

1.4. ANALYTICAL MODELS

Analytical models try to recreate the atmosphere by using one big mathematical formula. This has an advantage that it can be very performant and very flexible cause of the parametrization. But also has drawbacks in that most can only be used for ground view and not orbital view.

1.4.1. ISOTROPIC MODEL

This is the simplest model and is the one where newer and more accurate models are built.

1.4.2. PEREZ ET AL. MODEL

This model was released in 1993 and is seen as a very accurate model. This introduced a lot of parameters to the function to increase correctness and controllability. A slightly adjusted version in [oIC 04] which is the model presented by CIE, the international commission on illumination and approved by ISO, International Organization for Standardization.

1.4.3. PREETHAM ET AL. MODEL

This model was released in 1999 and is an improvement on the Perez et al. Model. It introduces the term turbidity – (How hazy the sky is) into the equation to help increase the accuracy and flexibility of the model. It also analytically calculates the parameters used in the Perez formula instead of using data tables. They did this by non-linearly optimizing the parameters to fit reference images generated with the Nishita et al. model.

1.4.4. HOSEK AND WIKIE MODEL

This model was released in 2012 and claims to be an improvement on the Perez et al. and Preetham et al. models as seen in *Figure 3*. The model is very similar to the Perez et al. model because it uses almost the same formula. Hosek and Wikie elaborated on the Perez et al. model by adding an anisotropic term and more parameters to give more control and add accuracy compared to the Perez et al. model. [Kol 12] investigates the claims of Hosek and Wikie. Comparing it to the Preetham et al. model also an improved Perez et al. model. Thus, concluding that indeed the Hosek et al. model is more accurate, certainly at low turbidity where the Preetham et al. model is 2-5 times too bright according to [Zotti 07]. Who compared [Preetham 99] with [oIC 04]. [Kol 12] further compares the performance of both models and concludes that indeed that [Hosek 12] is negligibly slower compared to [Preetham 99]. However, an optimized version of [Preetham 99] does optimize the performance by 30%. [Kol 12] also concluded that the solar radiance in the Hosek et al. model was not correct for lower solar angles. However, in 2013 the model was updated with a solar radiance function which corrected this problem. This leaves open for discussion which model is better, it is heavily dependent on the requirements of the project performance or accuracy. Later in 2013 the model was updated again to accommodate for use in a binary system like Tatooine from Star Wars.

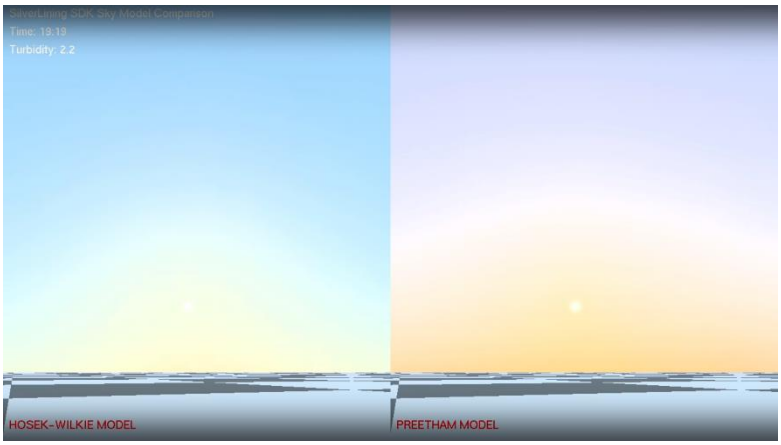

Figure 3: Comparison between Hosek et al. model and Preetham et al. model

This model seemed to be the best choice for this project and thus was chosen. Since it has the same performance as the Preetham model and is more accurate. It also is proven to work in a binary system and with different star types.

1.5. PRECOMPUTED MODELS

These models strive for efficiency and realism, so they try to use all the computational power of a computer and its graphics card. To do this they try to precompute as much as possible and not approximate to much of the equation.

1.5.1. NISHITA ET AL. MODEL

Released in 1993 and improved in 1996 it is the first model that takes in to account the color of the light and atmosphere. The 1993 version uses single scattering while the 1996 version implemented multiple scattering.

1.5.2. BRUNETON ET AL. MODEL

This model was released in 2008, it considers Rayleigh and Mie multiple scattering. This model works for ground view as well as orbital view which makes it very polyvalent. It even has the ability to create light shafts. It is a very accurate and performant model, it is achieved by precomputing the light transport equation for al viewpoints, view directions and sun directions. This model tries to use the most out of the GPU capabilities by using 4D tables to store the precomputations.

1.6. SPECTRAL MODELS

These models are still too not performant enough to be consider for real time application.

2. EFFECTS OF A BINARY STAR SYSTEM ON PLANET

2.1. BINARY STAR SYSTEM

A binary star system is a star system with 2 stars either with 2 similar or different stars. These stars would generally be orbiting around a central point. This means that the center of this solar system is not a star but rather the space between the 2 stars. This also drastically changes the gravity the planet would experience cause 2 stars that have around the same light radiation as our sun would only be about 15% smaller than our sun. This would mean the earth has 2 stars of 85% the size of its current sun and these together would have 70% more mass than the current sun. this would affect the planet in many ways. First, our year would only be 280 days instead of 365. This is cause our orbit would be drastically smaller. Also, because the earth is closer to these suns they would look bigger than our current one. These suns would also affect our moon, it would also orbit our planet faster. Tides would also be affected in that they would be way more powerful cause of the increased gravity the planet would experience.

2.2. EFFECT ON LIGHT AND SHADOWS

Luckily light is additive so there wouldn't be much different on first sight. Depending on the size of the stars things might get brighter with bigger stars or darker with smaller stars. A much more apparent effect of the 2 stars is that there would be 2 shadows if the stars are far enough apart it would be apparent.

2.3. EFFECT ON THE ATMOSPHERE

2 stars would mean that the atmosphere wouldn't change super much. However, there are some small changes. First, the effect around the sun would be double and brighter on the overlapping parts between the stars. Also, the color of the sky might change depending on the position of the stars. This is caused if one is setting and the other one is still in the sky this might make the sky purple or different shades between blue

and red. For a more extended period compared to now. It would also make sunsets interesting cause no two would be alike.

2.4. ECLISPSES OF ONE STAR BY THE OTHER STAR

Cause the suns would be very close to each other they would eclipse each other every 5 days. This would reduce the received energy on earth by 30-40%. However, this would not affect the climate because it would only be a period of 6 hours which would be smoothed out by the other days.

2.5. EFFECT ON THE HABITIBILITY

Luckily if the planet is 4 times as far away from the suns as the distance between the suns, than the planet would have a normal unaffected orbit. As said before the climate would be unaffected. However, this irregularity with the suns' luminance could affect plant growth and o2 production.

3. EFFECTS OF DIFFERENT STAR TYPES

3.1. DIFFERENT STAR TYPES

In our universe there are many different star types. This goes from white dwarfs to blue giants and everything in between. There are many special and different types of stars but most commonly there are 7 classes. These are classified according to surface temperature ranging from 30.000k to 2.000k.

3.2. EFFECT ON LIGHT AND SHADOW

Depending on the color of the star everything could look different. This will affect the color of everything and if the star is reddish it will look like you have reddish sunglasses on. Even the shadows would be affected by the color.

3.3. EFFECT ON THE ATMOSPHERE

Cause there is light color difference the light will be scattered differently in the atmosphere thus the sky has a different color. This also might result in an overall brightness change of the sky.

Figure 4: This picture shows the difference between earth (left) with or normal sun and Proxima b (right) that has a Red Dwarf as star.

3.4. EFFECT ON HABITIBILITY

Since light color is heavily dependent on temperature every star has its own habitable zone. For brighter stars this will be further always, thus the sun in the sky will be smaller. For less brighter stars they will be closer and thus the sun will be larger in the sky. Different light colors can also affect the color of vegetation. This might cause less or more plants to be edible.

CASE STUDY

1. INTRODUCTION

To recreate the dynamic day and night cycle for a binary star system the Hosek-Wilkie model was Chosen. To implement this model into the Unreal engine it was opted to use only blueprints so it would be easy to implement for anyone that would like to use it. A C++ Version for Unreal was also made as well as a Unity prototype. To create a proof of concept, a day night cycle for binary star system was implemented in the already existing Unreal sky model.

2. IMPLEMENTING THE HOSEK ET AL. MODEL WITH BLUEPRINTS

This implementation is based on a C++ version of the model. This implementation was purely used as a reference cause of the complexity of the model and the little amount of info available about how to calculation of the parameters is conducted. It was based on the original C code Version created by Hosek et al..

2.1. CREATING THE SHADER

To create a shader in Unreal a Material had to be used. This meant that all the math had to be done with a graph. Using a graph for math however is not the easiest way to do it. Cause short equations get very messy very fast. As seen in *Figure 5* a small mistake easily goes unnoticed. This causes delays and unforeseen results because mistakes were overlooked. The shader for the Hosek et al. model consists of the modified Perez formula this is made up out of the original Perez formula, an anisotropic term and some extra parameters. These help increase control and accuracy. The anisotropic term has a very similar effect to Mie scattering and causes a localized glow around the sun. The light direction and camera direction are used to calculate what part of the sky the camera is looking at and how it should look.

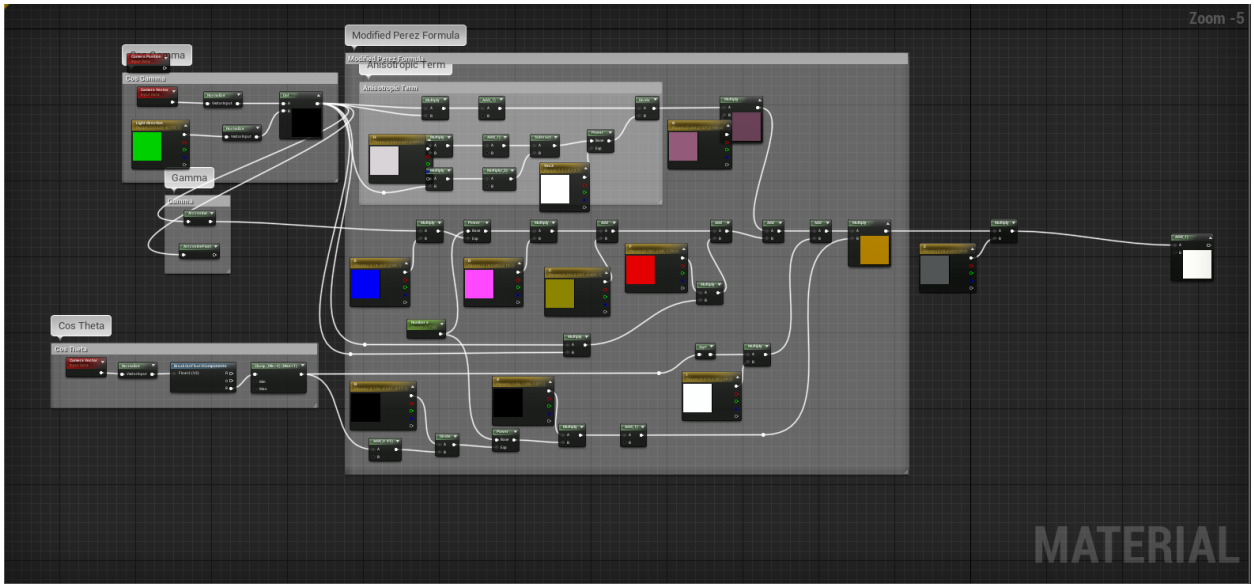

Figure 5: Hosek et al. Model Graph Shader Unreal

Once done with creating the shader there was no real way to test it as a direct result of there being no test data for the 10 parameters used in this formula. Also, there is no way to debug a shader so looking into the shader and seeing at what node it goes wrong at is no option. So to try out the shader I had to calculate the parameters.

2.2. CALCULATING THE PARAMETERS USED IN THE SHADER

To calculate these parameters, the Hosek et al. model uses datasets. These are basically arrays filled with values. Creating arrays in unreal blueprints isn't impossible however these datasets consists of 1080 values each so manual entry was no option. In Unreal there is the option to use data tables these can be seen as excel spreadsheets as seen in *Figure 6*. These data tables then were used to calculate the parameters using functions in blueprints.

2.2.1 USE OF DATA TABLES

These data tables describe splines who are then evaluated and used in the calculations. Parameters such as ground albedo and turbidity define which spline in the data table is used. Each parameter that is used in the shader is individually calculated according to a different spline.

	DataSet1	DataSet2	DataSet3
0	-1.100000	-1.140000	-1.370000
1	-0.134000	-0.198000	-0.491000
2	-4.080000	-7.510000	-41.000000
3	5.920000	8.400000	41.200001
4	-0.110000	-0.057000	-0.007390
5	1.600000	0.902000	0.484000
6	-0.000001	0.033900	0.006470
7	4.920000	4.770000	3.470000
8	0.513000	0.511000	0.509000
9	-1.170000	-1.170000	-1.520000
10	-0.183000	-0.185000	-0.650000
11	0.969000	2.960000	6.250000
12	0.095000	-2.260000	-5.660000
13	-0.047400	-0.157000	-0.019100
14	0.219000	0.634000	0.551000
15	0.110000	0.049800	-0.000022
16	3.600000	7.240000	2.510000
17	0.382000	0.422000	0.434000

Figure 6: Hosek et al. Model Data Table Unreal

2.2.2 EVALUATION OF THE SPLINES AND CALCULATING THE PARAMETERS

Using the turbidity, a spline start is calculated and used with the data tables. The Elevation of the sun is then used to calculate a single value. This is done 4 times using different starting positions. These 4 values are then used to calculate a single value which is then stored in the parameter. This whole process is then repeated 3 times because the parameters consist of 3 floats.

2.2.3 ENCOUNTERED ISSUES

As with the shader calculating the parameters was done using a graph. This gave way for the same problems of being very unreadable and confusing and prone to mistakes fast. *Figure 7* is an example of how large a simple equation can get. The bottem highlighted nodes are 1 equation consisting of simple aditions, multiplications , etc… . However, one wrong connection or missed connection means that a unforseen result is created. Finding these can be very hard as this is only one of 6 functions used to calculate the parameters.

A problem with the data tables was that the reference C++ version used float pointers to access its datasets. However, these are not possible in Unreal so the whole implementation had to be adapted to the use of data tables and the actual position in the table.

Evaluating the splines 3 times per parameter in different datasets caused another problem. Where, a column indicator had to be added cause the datasets were not stored in an array but in one data table. This required a small adjustment in the whole implementation.

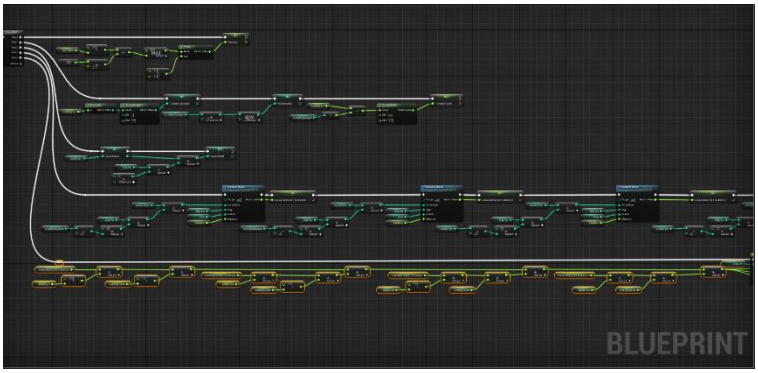

Figure 7: Hosek et al. Model Blueprint parameter calculation.

2.3 THE RESULT

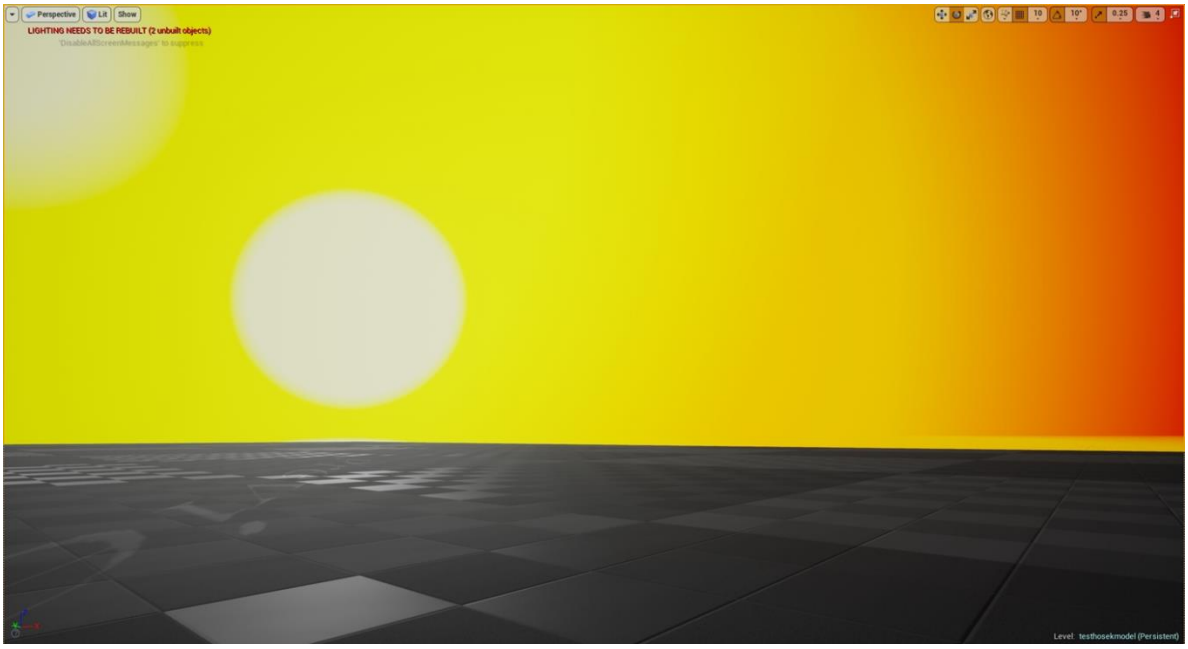

Figure 8: Hosek et al. Model Blueprint first result.

Clearly *Figure 8* is not what was expected as a result. There are some recognizable features present such as the sun which is to large and a certain "glow" around the sun. This means one of two things, either there is a problem in the shader or there is a problem calculation the parameters. However, there is no easy way of determining where the problem is. As stated before the shader can't be debugged nor are there test values with a known result to test the shader with. Also, no test values are available for the parameter calculations, so these can't be checked either. This meant another way which had to be found to check in what part things went south.

3. IMPLEMENTING THE HOSEK ET AL. MODEL WITH C++

To try and find the problem in the blueprint version an implementation of the reference C++ version in Unreal C++ was made. This was done purely to solve problems currently established in the implementation.

Checking the Parameter calculations means the shader should be unaffected and needs to be left just as is to compare with blueprint version.

As easy as it would seem to add already existing code to Unreal, it was not. Cause Unreal C++ has different rules from regular C++

3.1. CALCULATING THE PARAMETERS USED IN THE SHADER

A first problem that occurs here is that all the calculations and data sets are using doubles. However, a double is not allowed in Unreal, only floats are available. This meant that we had to change all the doubles in floats and make sure everything still worked. Another problem that arose is that the data sets used double pointers

to describe the position in the array of a value. Changing doubles to floats in this case did not solve the problem. Cause float pointers are not allowed in Unreal C++. This means a solution had to be found again for the starting position of the splines. The use of shared pointers instead of raw pointers didn't fix the problem. There was the option of using the data tables again. However, the addition of a spline start and dataset number could be used. This solution resembles the one used in blueprint although the raw datasets were used instead of the data tables.

3.2 THE RESULT

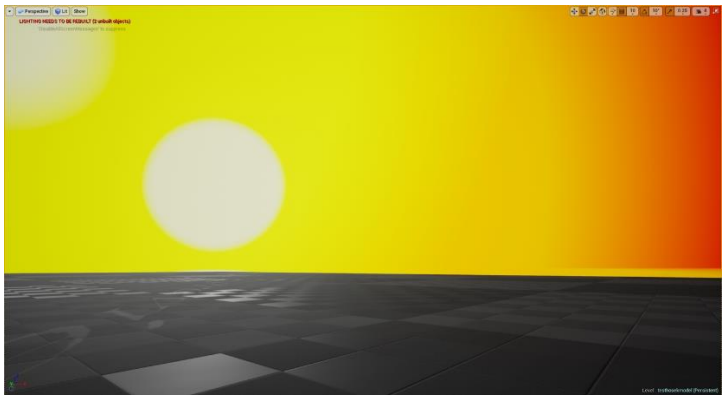

Figure 9: Hosek et al. Model C++ with blueprint shader result.

Figure 9 is the same as *Figure 8* which leads to the conclusion that the problem is not with the parameter calculation but with the shader. To make sure this was not a coincidence, a check was done to compare calculated parameters for the exact same circumstances. This revealed that indeed the returned parameters were identical up to 6 decimals (the lowest Unreal goes). Cause a lot of adjustments were made to the original C++ reference code there is still the option mistakes were made. This system works with a very small number often lower than 6 decimals so problems could be stemming from numbers getting rounded to 6 decimals. This means that we can't rule out the parameter calculations as the problem, although we can be 90% sure.

4. FIXING THE GRAPH SHADER

Now that the problem is situated in the shader the focus can be on fixing it. This is no easy task because as stated before, shaders aren't debuggable. Because of that stepping trough the graph in the material and seeing where the problem is, is not an option. This leave the option of trial and error or using hlsl in Unreal.

To make sure the reference shader wasn't flawed it was compared to a lot of different ones. This confirmed that the reference was similar to others and should have the same result. A slight problem is that all available reference shaders were made in glsl. This meant a conversion from glsl to hlsl had to be done. FX Composer was used to check for hlsl coding mistakes and syntax convention.

Now to use this shader in Unreal we must use a custom node in Unreal material editor, the problem here is that you can't use functions in this and Unreal does not tell you if there is a problem with the hlsl code. This meant that everything had to go in a single main function and be split up to find the problems.

```
float3 V = normalize(CamDirection);

float cos_theta = clamp(-V.z, 0.f, 1.f);
float cos_gamma = dot(V, SunDirection);
float gamma = acos(cos_gamma);

float3 t = 2.f* cos_gamma * H;

float3 te = float3(1.f,1.f,1.f) + H * H - t;

float3 ts = pow(te, float3(1.5f,1.5f,1.5f));

float3 chi = (float3(1.f,1.f,1.f) + cos_gamma *
cos_gamma) / ts;

float3 perez = ((float3(1.f,1.f,1.f) + A * exp(B / (cos_theta +
0.01))) * (C + D * exp(E * gamma) + F * (cos_gamma *
cos_gamma) + G * chi + I * sqrt(cos_theta)));

float3 R = Z * perez;
if (cos_gamma > 0.f) {
   R = R + pow(cos_gamma, float3(256.f,256.f,256.f)) * 0.5;
}

float4 color = float4(R,1.0);
   return color;
```

Figure 10: Hosek et al. Model HLSL shader.

So now the hlsl shader is implemented we can have a look at the result since this is the hlsl version of a working glsl server it would seem logical that the shader should be fixed now.

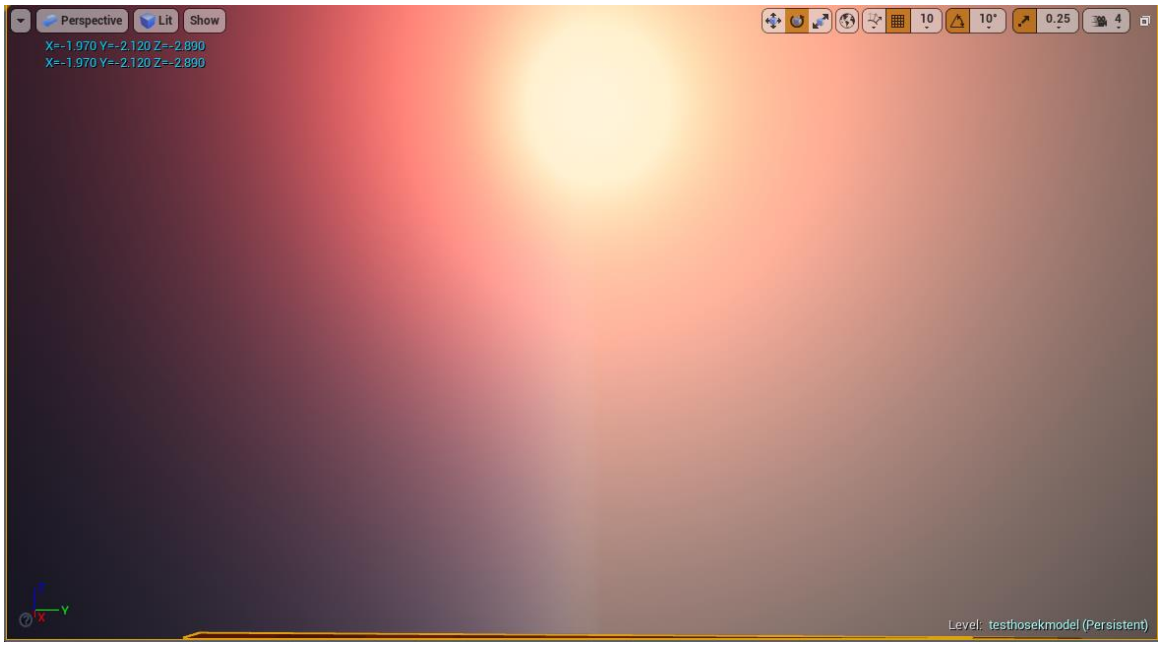

Figure 11: Hosek et al. Model HLSL shader first result.

Nothing is less true however there is visible improvement some new issues did arise. In *Figure 11* it is clearly visible that we now have a blue sky. Another improvement is the sun's glow it look more realistic although it is red now instead of its natural color. A big new problem is that the horizon looks to be vertical.

4.1 FIXING THE HLSL SHADER

The simplest problem to solve is the not horizontal horizon, this is due to reference shader being made for OpenGL which has different axis compared to unreal. In *figure 12* you can see the result with the correct horizon.

Figure 12: Hosek et al. Model HLSL shader fixed horizon result.

However, this doesn't solve the rest of the visible issues. After a lot of testing and trial and error there still is no improvement even though it feels like the implementation is so close to being finished it seems to be too far. An implementation using rust code was used as a third reference. This confirmed that parameter calculations were correct. The result however was very different for the same input visible if you compare *figure 12 and 13*.

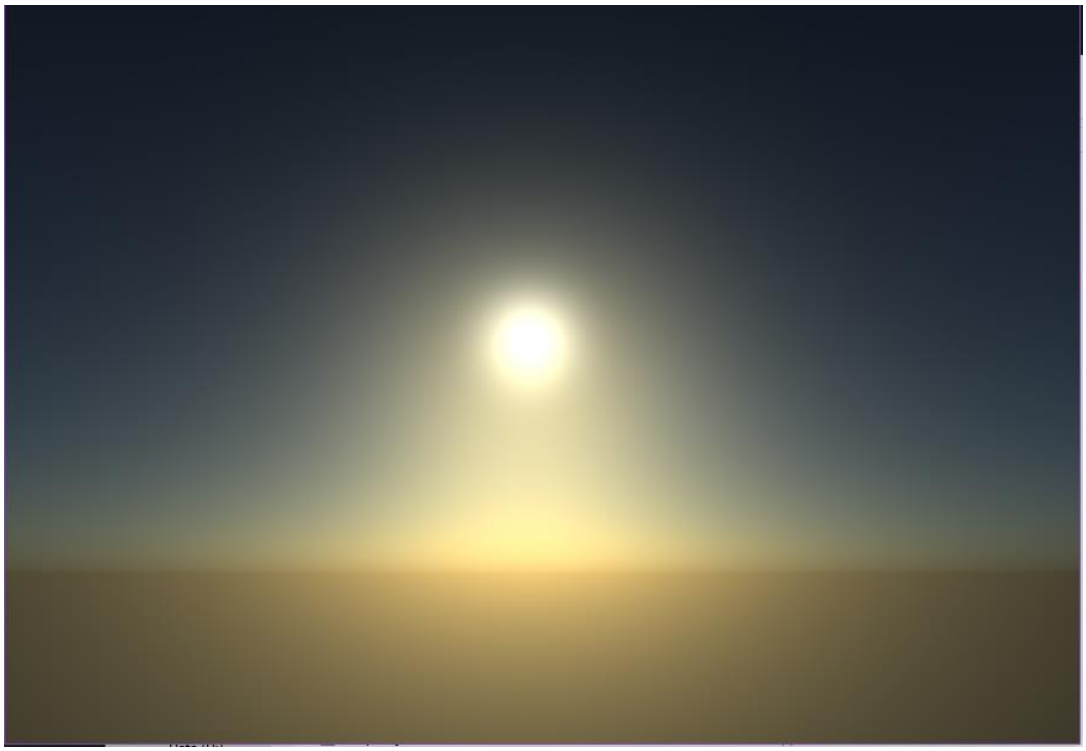

Figure 13: Hosek et al. Model rust version.

5. BINARY STAR SYSTEM IN UNREAL

Since the Hosek et al. model failed to implement correctly in Unreal. A basic and barebones version of the end result was produced using the existing Unreal Sky Model. This implementation simulates the movement of the binary stars in sky. It has a variety of adjustable options to adjust the 2 suns and their path in the sky.

5.1 ADDING SECOND SUN TO ALREADY EXISTING UNREAL MODEL

Adding the second sun to the existing unreal model wasn't too difficult. Merely a duplication of the shader was made that took care of the second sun. As seen in *figure 14* the sky behind the suns is very basic and inaccurate. It does however slightly affect the sunsets realistically by making them more purplish.

Figure 14: Binary Starsytem in Unreal 2 suns that orbit each other.

Further the implementation simulates this binary star system's orbit using seasons which include long and short days. It is capable of using different latitudes so that a different place on the planet can be simulated. The sun size and color can be adjusted. Ideally this would all be controlled by the temperature of the suns however this only works for the color as this is a simple version. The year length can be adjusted, ideally according to distance to the suns. The length of the day in simulation can be adjusted so an actual 24h day could be simulated in one second or 24 real life hours. This is just a small version of what would have been the result using the Hosek et al. model of course.

CONCLUSION

Binary star system has a lot of effect on a planet and would make a very special display in the sky depending on the star types. The sky could look a lot different in many combinations. Years could be longer or shorter days could seem longer. Sunsets would be a new spectacle every day. Sadly enough, this did not reflect on the end result although the potential of the Hosek et al. model is visible.

The Hosek et al. model seemed like the way to go when starting on this project. This is because it was performant yet accurate and was proven to work with binary stars as well as different star types. However, over the course of this project many obstacles arose, some were not concurred. This has left the end result to be desired as it is shows potential but isn't ready yet.

Having a livable planet in a binary star system may seem like something form a dream but in reality, it is very possible. This would create new challenges for the end result, because this would very much limit some options. Everything in the end result could be linked with the sun's temperature and the distance to the planet at its core. One small change could affect so many other things. Like sharing the size of one the suns so its smaller shortens the year and puts the sun closer so it appears larger. Yet makes the planet colder and darker.

A future addition could be an analytical night sky model to use in combination with the Hosek et al. model. This could create a very realistic overall feel to the cycle.

REFERENCES

203057.pdf. (n.d.). Retrieved from http://publications.lib.chalmers.se/records/fulltext/203057/203057.pdf

A Physically-Based Nightsky Model. (n.d.). Retrieved October 6, 2018, from http://graphics.stanford.edu/~henrik/papers/nightsky/

A Potentially Habitable World in Our Nearest Star - Planetary Habitability Laboratory @ UPR Arecibo. (n.d.). Retrieved October 6, 2018, from http://phl.upr.edu/press-releases/proxb

A Practical Analytic Model for Daylight. (n.d.). Retrieved October 6, 2018, from http://www.cs.utah.edu/~shirley/papers/sunsky/

Atmospheric Scattering. (n.d.). Retrieved October 15, 2018, from http://www.severewx.com/Radiation/scattering.html

Blue Sky and Rayleigh Scattering. (n.d.). Retrieved October 15, 2018, from http://hyperphysics.phy-astr.gsu.edu/hbase/atmos/blusky.html

Bouthors, A., Neyret, F., Max, N., Bruneton, E., & Crassin, C. (2008). Interactive multiple anisotropic scattering in clouds. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games - SI3D '08* (p. 173). Redwood City, California: ACM Press. https://doi.org/10.1145/1342250.1342277

Bruneton, E., & Neyret, F. (2008). Precomputed Atmospheric Scattering. *Computer Graphics Forum*, *27*(4), 1079–1086. https://doi.org/10.1111/j.1467-8659.2008.01245.x

Darula, S., & Kittler, R. (2002). CIE general sky standard defining luminance distributions.

Eric Bruneton. (n.d.). Retrieved September 25, 2018, from http://www-evasion.imag.fr/people/Eric.Bruneton/

EST_2.3_class6.pdf. (n.d.). Retrieved from https://fenix.tecnico.ulisboa.pt/downloadFile/1126518382176431/EST_2.3_class6.pdf

GPU Gems. (n.d.). Retrieved October 6, 2018, from https://developer.nvidia.com/gpugems/GPUGems2/gpugems2_chapter16.html

Hošek, L., & Wilkie, A. (n.d.). An Analytic Model for Full Spectral Skydome Radiance, 19.

Jensen, H. W., Durand, F., Dorsey, J., Stark, M. M., Shirley, P., & Premože, S. (2001). A physically-based night sky model. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques - SIGGRAPH '01* (pp. 399–408). Not Known: ACM Press. https://doi.org/10.1145/383259.383306

mie.png (602×166). (n.d.). Retrieved November 6, 2018, from http://hyperphysics.phy-astr.gsu.edu/hbase/atmos/imgatm/mie.png

Project - Analytical sky simulation. (n.d.). Retrieved October 6, 2018, from http://timothykol.com/project.php?id=3

PV Performance Modeling Collaborative | Hay and Davies Sky Diffuse Model. (n.d.). Retrieved October 6, 2018, from https://pvpmc.sandia.gov/modeling-steps/1-weather-design-inputs/plane-of-array-poa-irradiance/calculating-poa-irradiance/poa-sky-diffuse/hay-sky-diffuse-model/

PV Performance Modeling Collaborative | Isotropic Sky Diffuse Model. (n.d.). Retrieved October 6, 2018, from https://pvpmc.sandia.gov/modeling-steps/1-weather-design-inputs/plane-of-array-poa-irradiance/calculating-poa-irradiance/poa-sky-diffuse/isotropic-sky-diffuse-model/

PV Performance Modeling Collaborative | Perez Sky Diffuse Model. (n.d.). Retrieved October 6, 2018, from https://pvpmc.sandia.gov/modeling-steps/1-weather-design-inputs/plane-of-array-poa-irradiance/calculating-poa-irradiance/poa-sky-diffuse/perez-sky-diffuse-model/

PV Performance Modeling Collaborative | Reindl Sky Diffuse Model. (n.d.). Retrieved October 6, 2018, from https://pvpmc.sandia.gov/modeling-steps/1-weather-design-inputs/plane-of-array-poa-irradiance/calculating-poa-irradiance/poa-sky-diffuse/reindl-sky-diffuse-model/

PV Performance Modeling Collaborative | Simple Sandia Sky Diffuse Model. (n.d.). Retrieved October 6, 2018, from https://pvpmc.sandia.gov/modeling-steps/1-weather-design-inputs/plane-of-array-poa-irradiance/calculating-poa-irradiance/poa-sky-diffuse/simple-sandia-sky-diffuse-model/

Quora. (n.d.). Could An Earth Like Planet Exist Around Two Suns? Retrieved October 22, 2018, from https://www.forbes.com/sites/quora/2018/02/02/could-an-earth-like-planet-exist-around-two-suns/

Rayleigh scattering. (2018). In *Wikipedia*. Retrieved from https://en.wikipedia.org/w/index.php?title=Rayleigh_scattering&oldid=862220237

Rayleigh scattering. (n.d.). Retrieved October 6, 2018, from http://www.philiplaven.com/p8b.html

Real-Time Atmospheric Scattering. (n.d.). Retrieved October 6, 2018, from https://www.gamedev.net/articles/programming/graphics/real-time-atmospheric-scattering-r2093/

Scater-Light.jpg (367×262). (n.d.). Retrieved November 6, 2018, from http://www.schoolvideos.in/wp-content/uploads/2016/01/Scater-Light.jpg

See a star that changes its brightness | EarthSky.org. (n.d.). Retrieved October 6, 2018, from http://earthsky.org/sky-archive/find-lyra-the-harps-variable-star

September 15, N. W. |, & ET, 2011 03:02pm. (n.d.). What Would Earth Be Like with Two Suns? Retrieved October 22, 2018, from https://www.livescience.com/33500-earth-two-suns-tatooine.html

Sky & Atmosphere. (n.d.). Retrieved October 6, 2018, from http://vterrain.org/Atmosphere/

Sperlhofer, S. (n.d.). Deferred Rendering of Planetary Terrains with Accurate Atmospheres, 122.

Stellar classification. (2018). In *Wikipedia*. Retrieved from https://en.wikipedia.org/w/index.php?title=Stellar_classification&oldid=861900940

Wilkie, A., & Hošek, L. (2013). Predicting Sky Dome Appearance on Earth-like Extrasolar Worlds. In *Spring Conference on Computer Graphics - SCCG '13* (pp. 145–152). Smolenice, Slovakia: ACM Press. https://doi.org/10.1145/2508244.2508263

Zucconi, A. (2017, October 10). The Mathematics of Rayleigh Scattering. Retrieved October 6, 2018, from https://www.alanzucconi.com/2017/10/10/atmospheric-scattering-3/