

UKRAINIAN CATHOLIC UNIVERSITY

BACHELOR THESIS

**Implementation and comparison of
different microfacet specular BRDFs in
Unreal Engine 5**

Author:
Zoryana HERMAN

Supervisor:
Yurii DIACHYSHYN

*A thesis submitted in fulfillment of the requirements
for the degree of Bachelor of Science*

in the

Department of Computer Sciences and Information Technologies
Faculty of Applied Sciences



APPLIED
SCIENCES
FACULTY ●

Lviv 2023

Declaration of Authorship

I, Zoryana HERMAN, declare that this thesis titled, "Implementation and comparison of different microfacet specular BRDFs in Unreal Engine 5" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

“The wonderful things in life are the things you do, not the things you have.”

Reinhold Messner

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Bachelor of Science

**Implementation and comparison of different microfacet specular BRDFs in
Unreal Engine 5**

by Zoryana HERMAN

Abstract

Specular reflection calculations are one of the essential tasks in computer graphics. Because of the importance of its visual factor and because of the computational complexity of such calculations, a lot of different approaches and optimization techniques were proposed and are popular throughout the industry. Despite the popularity of various methods, it is common for game engines to offer only one established approach for specular calculations. In this work, we discuss various popular specular BRDFs and integrate some of them in Unreal Engine 5 in the form of shading models. That way, we provide users with more than one way to calculate specular highlights. We also include visual and performance cost analyses of the models and elaborate on the results.

The modified Unreal Engine source code can be accessed via the link: <https://github.com/zoriankaH/UnrealEngine> (In order to gain access to this repository, one needs to first request access to the UE source code as is described in the documentation <https://docs.unrealengine.com/5.2/en-US/downloading-unreal-engine-source-code/>).

The UE project used in this thesis can be accessed via the link: <https://github.com/zoriankaH/Specular-Models-UE-Test-Project>.

Acknowledgements

I want to express gratitude to my supervisor Yuriy Diachyshyn for mentoring me throughout the development of this thesis. I want to express gratitude to all of the teachers who have been teaching and guiding me throughout my study years at the university. I also want to thank my family for supporting me. I want to thank the Armed Forces of Ukraine for making my graduation this year possible. Finally, I would like to thank my friends with whom I shared my journey at UCU.

Contents

| | |
|---|------------|
| Declaration of Authorship | i |
| Abstract | iii |
| Acknowledgements | iv |
| 1 Introduction | 1 |
| 2 Background Knowledge | 3 |
| 2.1 Physically Based Rendering | 3 |
| 2.2 Bidirectional Reflectance Distribution Function | 3 |
| 2.2.1 Physically Based BRDFs | 3 |
| 2.2.2 Specular and Diffuse Reflections | 4 |
| 2.2.3 Microfacet Theory | 4 |
| 2.2.4 Normalization of BRDF | 5 |
| 2.2.5 Anisotropic BRDFs | 5 |
| 3 Related Works | 6 |
| 3.1 Specular Calculations in Unreal Engine | 6 |
| 3.2 Specular Calculations in Pixar | 6 |
| 3.3 Specular Calculations at Disney | 7 |
| 3.4 Specular Calculations in Call Of Duty Production | 8 |
| 3.5 Other Approaches | 8 |
| 4 Method | 10 |
| 4.1 GGX Specular BRDF | 10 |
| 4.1.1 Specular D | 10 |
| 4.1.2 Specular G | 10 |
| 4.1.3 Specular F | 10 |
| 4.2 Blinn-Phong Specular BRDF | 11 |
| 4.2.1 Specular D | 11 |
| 4.2.2 Specular G | 12 |
| 4.3 Cook-Torrance Specular BRDF | 13 |
| 4.3.1 Specular D | 13 |
| 4.3.2 Specular G | 13 |
| 4.4 Burley's (Disney) Specular BRDF | 13 |
| 4.4.1 Specular D | 13 |
| 4.4.2 Specular G | 14 |
| 4.5 Integration of Custom Shading Models in Unreal Engine | 14 |
| 4.5.1 GBuffer | 14 |
| 4.5.2 HLSL Translator | 15 |
| 4.5.3 Deferred Light Pass | 16 |

| | |
|---|-----------|
| 5 Experiments | 17 |
| 5.1 Implementation Details | 17 |
| 5.2 Shaders and Materials | 17 |
| 5.3 Visual Comparison of Models | 18 |
| 5.4 Comparison of Models by Performance Costs | 20 |
| 6 Results | 24 |
| 6.1 Visual Difference | 24 |
| 6.1.1 Material Comparison | 24 |
| 6.1.2 Environment Comparison | 24 |
| 6.2 Performance Cost Difference | 25 |
| 7 Conclusions and Future Work | 26 |
| 7.1 Conclusions | 26 |
| 7.2 Future Work | 26 |
| Bibliography | 27 |

List of Figures

| | | |
|-----|---|----|
| 4.1 | Shading Model visualization feature in UE5. | 15 |
| 4.2 | Buffer visualization feature in UE5. | 15 |
| 5.1 | Properties of a material that uses Simple Lit shader. | 17 |
| 5.2 | Rows of spheres with described above materials and different shading models. From left to right: Blinn, Cook-Torrance, Burley and Default Lit (GGX) specular model. | 18 |
| 5.3 | Metal material with different shading models. From left to right, top to bottom: Default Lit (GGX), Burley, Blinn, Cook-Torrance specular model. | 19 |
| 5.4 | Rubber material with different shading models. From left to right, top to bottom: Default Lit (GGX), Burley, Blinn, Cook-Torrance specular model. | 19 |
| 5.5 | Solid black material with different shading models. From left to right, top to bottom: Default Lit (GGX), Burley, Blinn, Cook-Torrance specular model. | 20 |
| 5.6 | Old West scene overview with different models. From left to right, top to bottom: Default Lit (GGX), Burley, Blinn, Cook-Torrance specular model. | 21 |
| 5.7 | Polar facility scene overview with different models. From left to right, top to bottom: Default Lit (GGX), Burley, Blinn, Cook-Torrance specular model. | 21 |

List of Tables

| | | |
|-----|---|----|
| 5.1 | Milliseconds Per Frame comparison for different specular models of the Old West environment scene. | 22 |
| 5.2 | Milliseconds Per Frame comparison for different specular models of the scene full of metal statues. | 23 |

List of Abbreviations

| | |
|-------------|---|
| BRDF | B idirectional R eflectance D istribution F unction |
| UE | U nreal E ngine |
| PBR | P hysically B ased R endering |
| NaN | N ot a N umber |
| NDF | N ormalized D istribution F unction |
| RMSE | R oot M ean S quared E rror |
| HLSL | H igh- L evel S hader L anguage |
| MSPF | M illi S econds P er F rame |
| FPS | F rames P er S econd |
| GPU | G raphics P rocessing U nit |

Physical Constants

The constant Pi $\pi = 3.14159$ (approximation)

To my family

Chapter 1

Introduction

In computer graphics, particularly within animated movie production and game industries, one of the most crucial problems was always light and shading rendering techniques. It is a matter of the performance cost as well as a factor that considerably influences the overall look of the product. With further developments within the industry regarding the topic, physically based rendering became very popular. Nowadays, it is the most used technique for shading calculations. One of the essential elements that should be included in those calculations in order to be physically based is Bidirectional Reflectance Distribution Function or BRDF. Generally, BRDF describes the way a surface reflects light. Light reflections are composed of diffuse and specular components. To put it briefly, the diffuse component describes the part of the reflected light that scatters into the surface and appears as a softer reflection, particularly seen on rougher surfaces. The specular component is associated with more intense and mirror-like reflections. The two factors are usually calculated separately and later combined to create a uniform result. From the detailed research regarding this topic, we concluded that specular components are usually more debatable among the two, and many engineers use either completely their own or different from others' approaches. This is partially because, unlike the diffuse component, specular reflection depends on and changes with the viewer's location.

There are a few ways to calculate specular BRDF. One of the popular methods was introduced by Cook and Torrance (1982), and it was based on the microfacet theory. The main factor that the calculations of such models depend on is the microfacet distribution function. As there is no definite method of calculation of this function and a lot of approaches were proposed throughout time, game and animation engine creators have to face the problem of choosing the most suitable calculations for their purpose.

The game engine that is used for this thesis, Unreal Engine, is an open-source engine and is widely used in the game industry by a number of game studios. However, because of the way the PBR shading works in Unreal, its users are bound to use shading calculations that were, to some extent, primarily integrated into the engine by the Unreal Engine creators. Nevertheless, Unreal Engine provides users with the ability to create their own shaders and to base them on various available by default shading models. Indeed, in Unreal Engine 5, there are twelve shading models to choose from, and some of them are designed to be used for specific cases like the modeling of hair and foliage. However, even with the use of different models and shaders, one will not be able to influence the actual shading calculations, and more in particular, specular reflection calculations, which are almost the same for every default UE shading model.

This can be a problem if the game creators are aiming for a specific stylized look which can be achieved only by changing the specular component. That is the reason why, in this work, we decided to implement and add a few additional shading

models into the engine that will have different from the default specular component calculations.

In the next Chapter [2](#), we provide basic information about the topic of specular calculations. Then in Chapter [3](#), we review the methods and works on different specular models that are used in popular movie production and game engines. Based on this research, we pick a few models that are common and then in-depth explain the calculations behind each of them. In Chapter [4](#), we describe the implementation and integration of those models into the engine. In Chapters [5](#) and [6](#), we compare the usage of different models in Unreal Engine both visually and by performance cost. Lastly, in Chapter [7](#), we provide conclusions and discuss possible improvements.

Chapter 2

Background Knowledge

2.1 Physically Based Rendering

As was mentioned in the previous chapter, rendering materials and light reflections were always important problems in the computer graphics field. Over the years, a great number of different shading models have been proposed, and they generally can be divided into two categories: empirical and physically based models. PBR became more and more popular over time as it has many advantages. If the game creators do not aim for a photorealistic look, it may seem that making physically plausible models is too expensive and unnecessary. However, it is not always the case. For starters, making the core of shading models physically based allows one to easily utilize this model for different shaders and materials by changing up the parameters. If the model is empirical, it is harder to experiment with. Empirical models also do not have a real-life standard to follow, unlike physically-based ones, so if there is an issue with the model, it may be harder to allocate it. Another point that Gotanda, Kawase, and Kakimoto (2015) describe in their paper is that the judgment of artists is not typically the best measure for choosing the shading parameters, and it is better sometimes to limit their possible influence on shading calculations and make shading parameters correspond to physical units. This is again easier to achieve using PBR.

2.2 Bidirectional Reflectance Distribution Function

An important element of shading calculations is light reflections. The function that describes how light reflects off a surface is called Bidirectional Reflectance Distribution Function, or BRDF. BRDF takes ingoing and outgoing light direction as well as some additional parameters that influence the reflectiveness of the surface.

2.2.1 Physically Based BRDFs

In order to be physically based, there are a few rules that BRDF should satisfy.

The first one is Helmholtz reciprocity, and it accounts for the function being bidirectional, meaning the incoming and outgoing light directions can be swapped places, and this should not influence the outcome, including the reflected energy.

Another rule is energy conservation, which states that if a surface is perfectly non-absorbent and its albedo is fully white, the amount of reflected energy should be equal to the amount of received energy. There is a relevant study by Boksansky (2021) where he compares different specular BRDFs. He describes in this work that, in practice, models are sometimes not perfectly energy-conserving because as long as the reflected energy is less or equal to the received, it should not stand out too much visually.

The last rule is the positivity of BRDF, meaning it should always be greater than 0.

2.2.2 Specular and Diffuse Reflections

The overall reflection is usually calculated by combining the diffuse and specular BRDFs. The diffuse part corresponds to the light that at first scatters into the surface and later exits the material, reflecting under different from the incoming angle. Rougher surfaces are usually associated with diffuse reflections. Specular reflection, on the other hand, is part of the light that is reflected away upon hitting the surface at the same angle. Smoother surfaces, like perfect mirrors, reflect only specular types of reflections. In this work, we focus on the specular part of the calculations.

2.2.3 Microfacet Theory

One of the popular ways to calculate specular BRDF is to follow the microfacet theory that was introduced by Cook and Torrance (1982) in their paper. When taking the microfacet approach, we assume that the surface is made out of a series of microfacets facing different directions. We assume that only the microfacets whose normals are equal to the half-angle vector (meaning the vector that bisects the angle between the view and the light direction vectors) contribute to the specular reflection. This is the model that will be used as the core base of all models implemented in this thesis, and its corresponding BRDF is defined as:

$$R(l, v) = \frac{D(h)F(v, h)G(l, v, h)}{4(n \cdot l)(n \cdot v)}$$

Where:

- l - vector that points in a direction of a light source
- v - view vector that points in the direction of a viewer
- h - half-way vector, lies in between l and v vectors
- n - surface normal vector

D here is a normalized microfacet distribution function, and it describes the part of microfacets that are facing in the H direction and therefore contribute to the specular lobe. In other words, NDF is the probability density function that the normal of the microfacet will be oriented towards the needed direction. The distribution function is the main factor that determines the look of the specular peak.

G here is geometric attenuation factor. G accounts for the part of microfacets in a sample that do not contribute to the specular lobe due to the mutual masking and shadowing. Meaning, if the light is reflected off the microfacet but on its way, is blocked by another microfacet we assume that that reflected light gets lost. It is observed that in real life, it works a little differently, as some of the light may bounce back multiple times and eventually contribute to the reflection, but it would be computationally expensive to account for such cases, so instead, we use approximations like geometric attenuation factors.

F here is a Fresnel term. Fresnel term accounts for the increased intensity of the reflections near the grazing angles (meaning when the incidence angle approaches 90 degrees). This effect can be vividly seen in the reflections of smoother surfaces like

water bodies. Fresnel term also accounts for the visual difference between metals and dielectrics.

The denominator in this formula comes from the derivation of the model and ensures its normalization.

One parameter in this formula differs from the parameter in the formula derived by Cook and Torrance - the constant 4 in the denominator. In the formula given by Cook and Torrance, there was π instead. There is an explanation on this matter in the paper from Walter et al. (2007), where he describes that the reason for that is that the normalization of the distribution function in Cook and Torrance's paper is different from how it is normalized nowadays for most models and that therefore it is correct to use the constant 4.

2.2.4 Normalization of BRDF

In general, normalization of BRDF is scaling of its intensity to the angular size. For the BRDF to be correctly normalized, it means that it should be integrated over the hemisphere. This accounts for the fact that the total amount of energy reflected from every direction should equal the total amount of received energy. Normalization of BRDF is also required for it to be correctly energy-conserving, as Sébastien Lagarde points out in his blog¹.

2.2.5 Anisotropic BRDFs

In computer graphics, it is common to characterize materials as either isotropic or anisotropic. The materials are considered anisotropic when the properties of materials change when measured in different directions. These properties can be physical, like strength. However, in terms of computer graphics, it is the shape of the specular highlight that changes. This way, if the material is anisotropic, depending on the view direction or the rotation and position of the object, the shape of the specular lobe changes. Some examples of anisotropic materials include hair, brushed metals and the surface of CDs. Because of their behaviour, BRDFs for anisotropic materials are different in calculations. Ward (1992) in his paper, proposed an anisotropic specular model based on Beckmann distribution. Ashikhmin and Shirley (2001) did the same in their paper but for the Phong model.

Unreal Engine 5 supports anisotropy for some of the shading models. Furthermore, the anisotropic GGX distribution-based BRDF that is used in Unreal for Default Lit (default shading model) anisotropic materials is based on the anisotropic model described in the article by Burley (2012). In this work, we do not cover anisotropy in detail and do not add support for it to the implemented models. This is due to the fact that most of the basic materials, such as metals or plastic, can be visually replicated using only isotropic models.

¹<https://seblagarde.wordpress.com/2011/08/17/hello-world/>

Chapter 3

Related Works

3.1 Specular Calculations in Unreal Engine

Since this thesis is conducted on Unreal Engine, we first researched how shading calculations are done in UE and what specular model is used. Karis (2013), an engineer in Epic Games, in his article about PBR in Unreal Engine, pointed out that the general choice of what specular model to use was made in favour of the GGX model. Some of the choices, for example, roughness parameter calculation, were also adopted from Disney's (Burley, 2012) model, which is discussed later in the thesis. As for the geometric attenuation factor, they have changed it in favour of geometric visibility factor V , which is a geometric attenuation factor divided by the denominator of the specular BRDF. The idea of using a visibility factor instead of geometric attenuation was described in the paper by Kelemen and Szirmay-Kalos (2001), and it is generally a good way to avoid NaN values and division by zero.

Because the paper by Karis (2013) was written about Unreal Engine 4, it is a little outdated, and some details do not match in the version 5 of UE, which is used for the implementation in this thesis. For example, in the paper, it is mentioned that they use the Schlick (1994) approximation of the G factor. However, after analyzing the source code, we concluded that instead, the optimized Smith version derived for GGX by Walter et al. (2007) is used. The Fresnel term is also different in this version from the one explained in the paper. Approximation of the Fresnel introduced by Schlick (1994) is still used. However, allegedly they no longer use the Spherical Gaussian approximation as the power and instead use the original power of 5, as it was introduced by Schlick.

Karis mentioned that the choice to use GGX was made primarily with the feedback from artists because the GGX model has a distinguishably longer specular lobe tail than those models which use Beckmann or Blinn NDFs. The preference for a longer specular tail is mentioned in several studies, and it appears that the final choice of the specular model often comes down solely to the artists' preferences and does not involve much debate over the performance cost differences between models.

3.2 Specular Calculations in Pixar

Movie production companies such as Pixar usually use offline (meaning not real-time) rendering to create their product. Such rendering technique is much less performance critical than real-time rendering and therefore enables the use of more shader parameters and the ability to provide artists with more choice regarding shading calculations. Pixar rendering pipeline, as described in the related paper by

Hery and Villemin (2013), provides a choice of multiple specular BRDFs. Furthermore, there are a few isotropic and a few anisotropic versions of specular BRDFs.

One of the models, and the one that is highlighted the most in the article, is Beckmann distribution-based BRDF introduced by Beckmann and Spizzichino (1963) in the relevant study. It is mentioned that this model was favoured due to its simplicity and efficiency. They also mention that it is similar to the Cook-Torrance model, which was the first introduced microfacet specular BRDF and is one of the models implemented in this thesis.

Pixar's BRDF is a little different to the traditional specular BRDF model in a way that they do not use explicit geometric attenuation factor and Fresnel factor. This accounts for the mentioned simplicity and efficiency of their BRDF.

One other factor that is referenced a lot in the explanations of the BRDF in the paper is the white furnace test. The white furnace test was introduced by Heitz (2014), who described that it is a good way to test if the BRDF is energy-conserving. The idea is to create a white sphere and illuminate it with light in all directions: if it conserves energy correctly, the sphere will appear fully white. If it appears otherwise, it would mean that energy either gets lost or is gained during calculations. Because of its shape, the sphere object is also good to use to help indicate the problem with energy conservation in case the sphere does not appear fully white.

3.3 Specular Calculations at Disney

Burley (2012) in his paper about physically based shading at Disney mentioned that in order to find the answers to the questions of which shading models to use, they conducted a study comparing different types of real-world materials and different specular models. He mentioned that a few popular choices that use Beckmann, Blinn-Phong (Blinn, 1977) and Gaussian distribution functions failed to satisfy their needs as were not realistic enough. He also points out that these three methods produce very similar results. GGX distribution, as it was derived by Walter et al. (2007), seemed to produce results the closest to what Disney was aiming for, and so was primarily chosen as their specular model for the same reason that was described by Karis (2013) - wider specular lobe tail. For the roughness value, they used the square value of material roughness, which was also adopted by UE. Although the GGX distribution was the closest to the desired result by Disney, they concluded that for some of the materials, the specular tail is still not long enough. Consequently, they evaluated a few more iterations of the same in its core distribution, including Berry's distribution which differs from GGX in that the exponent in Berry's form is 1 instead of 2. Considering the results of this experiment, they derived a new form of distribution function called Generalized-Trowbridge-Reitz that has a constant variable in the exponent. It is said that the optimal values for this variable are in a range [1, 2].

As for the geometric attenuation factor, they found that it has a significant impact on the directional albedo and how albedo changes near the grazing angles, which is a factor that represents the amount of energy reflected by the surface. This greatly impacts the overall look of the objects because surfaces with lower albedo are darker in colour and vice versa. Disney tested the albedo and the change of reflectance at grazing angles for multiple different geometric attenuation factors and found that when using the Smith term for GGX that was derived by Walter et al. (2007), the material gains extreme reflectance near the grazing angles for shiny surfaces, which is not natural for the real world materials. Due to that, they at first decided to use the G term by Walter but remap the roughness parameter to avoid that unwanted

shininess. Moreover, they linearly scaled the material roughness so that it would be in the range $[0.5, 1]$ before squaring it to fit the roughness parameter. However, later a “Specular G revision” was added to the main article where they mentioned that based on the paper by Heitz (2014), where different specular G approaches were analyzed in detail, they dropped their original remapping method and concluded that the previously mentioned problem of extreme shininess could have been a false conclusion due to the incorrectly measured data samples.

For the Fresnel equation, they decided to take the common approach of using the Schlick (1994) approximation.

3.4 Specular Calculations in Call Of Duty Production

Hoffman (2010), in the related paper as well as Lazarov (2011) in similar work describe physically based shading as well as specular calculations used in the development of Call of Duty games.

In both papers, the Blinn-Phong distribution is favoured. Lazarov, in his work, describes the use of Blinn-Phong over the Beckmann distribution as it is cheaper in terms of performance costs and “visually more intuitive”. Hoffman compares Phong with the Blinn-Phong distribution. He describes that Blinn’s approximation of the Phong specular model is actually more physically plausible and produces more accurate results. Although the model introduced by Phong (1975) was the original model, it works acceptably only for perfectly reflective surfaces such as chrome balls and fails to satisfy the reflections of rougher surfaces. Blinn (1977) proposed to change the reflection vector used in the Phong model to a half vector, which is the vector that microfacet normal needs to be facing in order to reflect light. The reflection vector used in the Phong model, unlike the half vector, does not have a logical explanation when it comes to microfacet distributions.

After the research, we concluded that Phong specular in its original form is hardly used nowadays. Consequently, we decided not to implement the original Phong specular shading model in this thesis and instead implement Blinn-Phong, which is still widely used for its efficiency.

As for the geometric term, both authors describe the use of the mentioned earlier geometric visibility factor. In paper by Hoffman (2010), it is described that implicit geometric attenuation, which results in a visibility factor of value 1, is preferred due to the lack of additional shading calculations. Although, the visual effect of this approach is not ideal because the falloff of the specular appears too abrupt and the reflection quickly becomes too dark. Additionally, both papers include analysis of multiple other terms, including Kelemen’s approximation of visibility function (Kelemen and Szirmay-Kalos, 2001) and Schlick-Smith’s (Schlick, 1994) approximation. However, it seems to be left to the debate if the use of these functions is worth the additional costs.

The original Fresnel factor in both papers is again substituted for approximation by Schlick (1994).

3.5 Other Approaches

There are a few more works that we decided are worth mentioning, however, will not be analyzed in such depth as the previous articles.

In Arnold Renderer, as is described by Langlands (2014), Cook-Torrance specular model is used.

Frostbite creators Lagarde and Rousiers (2014) used the GGX distribution function for their specular along with Smith height-correlated visibility function (which is the same function that is used in Unreal).

ILM studio (Snow, 2010) used the Cook-Torrance model-based specular in the production of Iron Man movies.

It appears that the choice of the specular model is not straightforward and that there are many different variations of specular components used within the same industries.

Chapter 4

Method

After the detailed research on the different specular models that are used in computer graphics, we picked three more models, besides the default GGX, that seemed to have the most visual difference and are widely used. In this chapter, we further elaborate on each model and the functions we used for the implementation of each of them.

4.1 GGX Specular BRDF

4.1.1 Specular D

GGX (also known as Trowbridge-Reitz) BRDF is the model that is used in Unreal Engine by default. GGX distribution was first derived by Trowbridge and Reitz (1975) in their paper. It was immediately favoured by different experts, including Blinn (1977), for the distinguishable long specular lobe tail. The formula for GGX NDF is as follows:

$$D_{GGX}(h) = \frac{a^2}{\pi((n \cdot h)^2(a^2 - 1) + 1)^2}$$

Where a is a roughness parameter that, in this case, is a square value of the surface roughness.

4.1.2 Specular G

Walter et al. (2007) derived Smith masking-shadowing function for GGX distribution. The visibility factor that was used in UE is described in the work by Heitz (2014). It is also similar to the one used in the Frostbite engine as described by Lagarde and Rousiers (2014). It is derived as an optimized height-correlated Smith term for GGX distribution. The formula for it is:

$$V_{Smith}(l, v, h) = \frac{n \cdot l(n \cdot v(1 - a) + a) + n \cdot l(n \cdot v(1 - a) + a)}{2}$$

It is important to note that the original formula for the height-correlated Smith is a little different. Although we could not find the exact same formula that is used in UE elsewhere in the literature, intuitively, its difference compared to the original is because of the further optimization.

4.1.3 Specular F

The Fresnel factor used in Unreal Engine is an approximation that was proposed by Schlick (1994). This approximation quickly became a popular choice to use instead

of the full Fresnel equation due to its simplicity and the small error that it produces. Not only is it cheaper than the original equation, but it also does not contain unintuitive parameters like the full formula, so it is far easier to tune for different cases, which is very important for physically based models. The formula for it is:

$$F_{Schlick}(v, h) = F_0 + (1 - F_0)(1 - (v \cdot h))^5$$

Where F_0 is the reflection at normal incidence. This value is dissimilar for metals and dielectrics.

There is an article by Hoffman (2010) where he concluded that for most cases, the Schlick approximation is actually even more precise than the original equation. Because of how popular the Schlick approximation of the Fresnel is and all the benefits that come with using it, we only use this approximation as a Fresnel term for all of the models and, therefore, will no further reference the choice of Fresnel factor in this work.

4.2 Blinn-Phong Specular BRDF

The Blinn-Phong specular model was proposed in the paper by Blinn (1977). He essentially proposed the optimization of the Phong (1975) model by substituting the reflection vector used in the Phong model for a half-vector. Although it can not be said that this model was originally fully based on microfacet theory like the Cook-Torrance model, Blinn actually mentions the microfacet concept in the paper. The use of a half-vector in the exponent also suggests a very similar approach to the microfacet-based models. Nowadays, it is common to see the use of Blinn-Phong distribution fitted into a microfacet approach, so this is how we are going to implement it.

4.2.1 Specular D

In order to be correctly normalized, the distribution function should be integrated over the hemisphere. Following this logic, the next equation should be satisfied:

$$\int_{\Omega} D(h)(n \cdot h)d\omega(\theta, \phi) = 1$$

Where $D(h)$ is the distribution function and $d\omega(\theta, \phi)$ is a solid angle element. This integral should then be iterated over every microfacet normal direction.

Using this equation, the normalized Blinn-Phong distribution function is as follows:

$$D_{Blinn}(h) = \frac{m + 2}{2\pi}(n \cdot h)^m$$

This is the normalized heightfield version of the distribution function, as is mentioned in the blog¹. This means that it normalizes the projected area of microfacets onto the macrosurface. Additional cosine ($n \cdot h$ dot product in this case) in the integral accounts for that.

m in this formula is the specular exponent of a surface. Specular exponents control the glossiness of surfaces and generally are different for different BRDFs. There is a paper by Sawicki (2021) in which different BRDFs and their corresponding glossiness coefficients are compared. He conducted the detailed analysis and,

¹<http://www.thetenthplanet.de/archives/255>

using MATLAB, approximated the conversion functions between the different coefficients that yielded the best results and provided the smallest RMSE. Following the results of this experiment, the best function to derive the specular exponent for the Blinn-Phong model using the roughness parameter used in GGX distribution in the default Unreal Engine model would be:

$$m = \frac{1.696^2}{a^2} - 4.5$$

Where a is the roughness coefficient of GGX BRDF.

However, upon further researching the topic of coefficient conversion, we found that it is far more common to use the conversion that Walter et al. (2007) described in his paper on Microfacet Models:

$$m = \frac{2}{a^2} - 2$$

This is also the conversion method that Karis uses in his description of Blinn distribution in the article in his blog². Because of that, we decided to use the latest formula.

4.2.2 Specular G

As for the geometric attenuation factor, the Smith integral has no closed-form solution for the Blinn-Phong distribution. Walter et al. (2007) mentioned that due to its similarity to the Beckmann distribution (in fact, for certain values, they are almost identical), it is best to use the geometric attenuation factor derived for Beckmann distribution:

$$c = \frac{n \cdot v}{a \sqrt{1 - (n \cdot v)^2}}$$

$$G_{Beckmann}(v) = \begin{cases} \frac{3.535c + 2.181c^2}{1 + 2.276c + 2.577c^2}, & \text{if } c < 1.6 \\ 1, & \text{otherwise} \end{cases} \quad (4.1)$$

However, for further optimization and because of the popularity of this approach, we use the approximation of the above formula that was introduced by Schlick (1994):

$$k = a \sqrt{\frac{2}{\pi}}$$

$$G_{Schlick}(v) = \frac{n \cdot v}{(n \cdot v)(1 - k) + k}$$

This is the same approximation that Karis (2013) describes using in UE 4. We multiply the above function that takes the view vector with the same function that takes the light vector to form a shadow-masking term as it was proposed by Smith (1967). We also divide this factor by the denominator of the BRDF equation to form Visibility Factor. The final formula for it is:

$$V_{Smith-Schlick}(l, v, h) = \frac{1}{4((n \cdot l)(1 - k) + k)((n \cdot v)(1 - k) + k)}$$

²<http://graphicrants.blogspot.com/2013/08/specular-brdf-reference.html>

4.3 Cook-Torrance Specular BRDF

Cook-Torrance BRDF was first introduced by Cook and Torrance (1982), and it was the first introduced microfacet specular BRDF. All of the models that are implemented in this work and that are described in Chapter 3 are based on the concepts introduced in this paper.

Cook and Torrance elaborated a lot on the topic of the colour of the specular lobe. Before, for the models like Phong, it was more so considered to match the colour of the light source. However, Cook and Torrance conclude that, with some exceptions, the colour of the specular reflection actually depends more on the colour of the material. This was also the main concept that helped them to better model the difference between metals and dielectrics, as the colour of the specular lobe for metallic surfaces depends more on the colour of the light source. They also discuss a lot the importance of using bidirectional reflection in order to simulate materials closely.

4.3.1 Specular D

As for the NDF, Cook and Torrance mainly discuss the Blinn and Beckmann distributions. However, they favour the latter as they concluded that it works nicely for various kinds of materials. The Beckmann NDF, as it was derived in the paper (Beckmann and Spizzichino (1963)), is:

$$D_{Beckmann}(h) = \frac{\exp\left(\frac{(n \cdot h)^2 - 1}{a^2(n \cdot h)^2}\right)}{\pi a^2 (n \cdot h)^4}$$

4.3.2 Specular G

For the self-shadowing term, Cook and Torrance describe using the V-cavity approach. The V-Cavity approach assumes that the surface is made out of a series of V-shaped cavities that vary in size. On the other hand, Smith's approach, which was used in the previous models, describes the surface made out of slopes. In Smith's approach, the visibility of microfacets does not matter. However, when using the V-cavity approach, we assume that only the microfacets that are visible, meaning have a visible cavity behind them, contribute to the specular highlight. Whether the cavity is visible is determined by the relation of the half-vector to the cavity angle. The formula for geometric attenuation factor as it is described by Cook and Torrance therefore is:

$$G_{Cook-Torrance}(l, v, h) = \min\left(1, \frac{2(n \cdot h)(n \cdot v)}{v \cdot h}, \frac{2(n \cdot h)(n \cdot l)}{v \cdot h}\right)$$

We again divide this function by the BRDF denominator to create a geometric visibility factor.

4.4 Burley's (Disney) Specular BRDF

4.4.1 Specular D

Burley's specular model, or Generalized-Trowbridge-Reitz, as it is named in the paper (Burley (2012)), is based on the GGX model. Disney wanted to provide the ability

to model a longer specular tail than GGX produced, so they derived the generalized version, which, unlike the GGX model, did not have a constant of 2 in the exponent of the distribution function, but instead the variable λ :

$$D_{Burley}(h) = \frac{(\lambda - 1)(a^2 - 1)}{\pi(1 - (a^2)^{1-\lambda})(1 + (a^2 - 1)(n \cdot h)^2)^\lambda}$$

In the paper, it is mentioned that the ultimate values for λ are between [1, 2]. For our purpose, we use the average value of 1.5.

4.4.2 Specular G

As for the geometric attenuation factor, they used Smith's factor derived for GGX by Walter et al. (2007). The visibility factor based on it is:

$$V_{Smith}(l, v, h) = (n \cdot v + \sqrt{n \cdot v(n \cdot v - n \cdot v * a^2) + a^2}) * (n \cdot l + \sqrt{n \cdot l(n \cdot l - n \cdot l * a^2) + a^2})$$

4.5 Integration of Custom Shading Models in Unreal Engine

Currently, there is little present material regarding the topic of integration of new shading models into the Unreal Engine. There is no documentation from Epic Games on the matter as well. There are a few related works on online forums^{3, 4, 5}, but some of them are outdated, as the shading rendering process changed noticeably with the introduction of version 5 of UE.

In order to add a new shading model to the engine, one needs to modify the source code in both C++ and shader files. The first part of this modification is done to ensure that the new model is visible on the editor level and, therefore, can be chosen to create new materials. This is done simply by adding additional enumerations and preprocessor directives to the source code. We also need to add some additional parameters, such as models' descriptions and debug colours, which can be used in the Buffer Visualization feature in UE (Fig 4.1).

4.5.1 GBuffer

Some of the modifications also need to be added to ensure that the model will be able to make use of GBuffer slots later on.

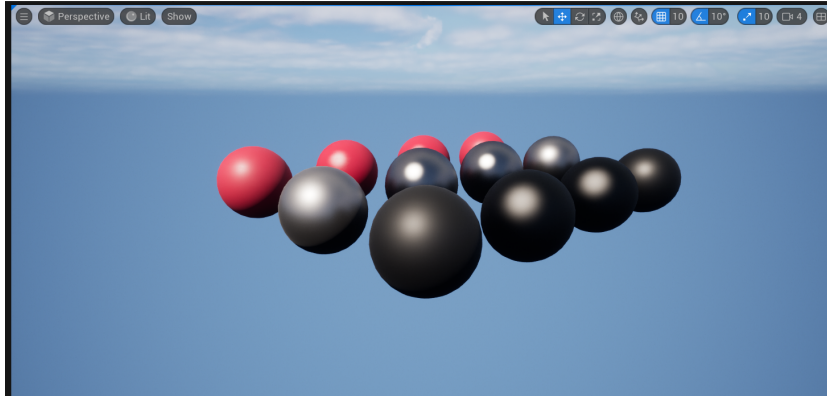
Gbuffer or Geometry Buffer is used in deferred shading and is responsible for storing information about image rendering. It consists of a series of render targets, including base colours, normals and metalness of the present geometry (Fig 4.2), that are combined after the lighting calculations to create a fully rendered image.

In order for our new model to be rendered correctly, we need to assign the GBuffer information regarding the shading model ID so that later on, when each pixel is rendered, it can access the information about what shading model to use. The GBuffer also needs to be fed the information regarding material parameters corresponding to the chosen model.

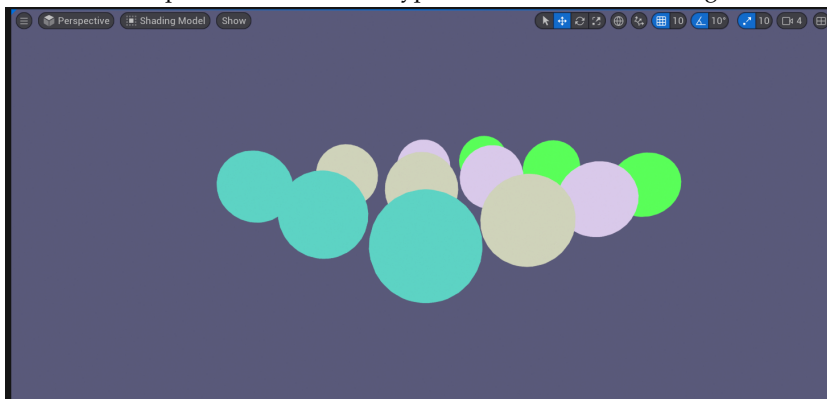
³<https://dev.epicgames.com/community/learning/tutorials/2R5x/unreal-engine-new-shading-models-and-changing-the-gbuffer>

⁴<https://medium.com/@lordned/ue4-rendering-part-6-adding-a-new-shading-model-e2972b40d72d>

⁵<https://medium.com/@solaslin/learning-unreal-engine-4-implement-cel-shading-w-o-utline-using-custom-shading-model-in-ue4-22-1-775bccdb9ffb>



(A) Scene view on spheres with different types of materials and shading models in UE5.



(B) Same scene as above but with Shading Model visualization feature turned on.

FIGURE 4.1: Shading Model visualization feature in UE5.

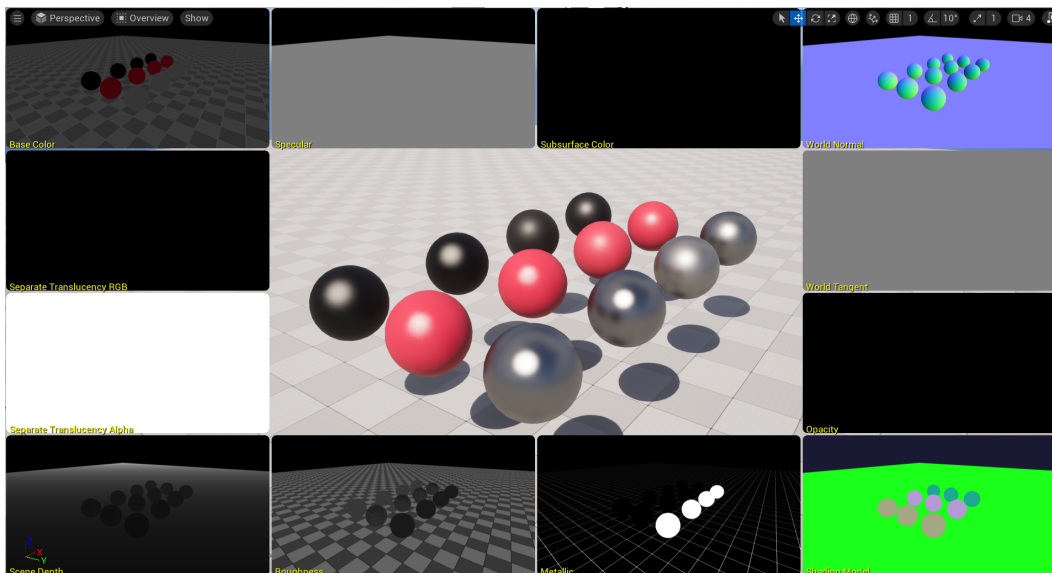


FIGURE 4.2: Buffer visualization feature in UE5.

4.5.2 HLSL Translator

After modifying the C++ part correspondingly, we need to add the new model to the HLSL translator. The HLSL translator is responsible for translating material graph

code to the HLSL, which is used to compile shaders for the needed material. Therefore defining the model to the translator makes it visible to the shader compiler.

We also need to enable the material slots that our model needs to have access to (such as Base Color, Metallic, and Roughness⁶).

4.5.3 Deferred Light Pass

Reflections, as well as all shading calculations, are calculated during the deferred light pass. The deferred light pass is executed after the base pass, during which the described before data is fed into GBuffer.

In order to correctly calculate the shading for our custom model, we need to define the cases when it is used and output BRDF that corresponds to it. In this work, the only difference between the shading calculations of all models is the specular component. The diffuse part, as well as model sampling, is based on the Default Lit shading model calculations.

⁶<https://docs.unrealengine.com/4.27/en-US/RenderingAndGraphics/Materials/PhysicallyBased/>

Chapter 5

Experiments

5.1 Implementation Details

Following the steps described in Section 4.5, we added three new shading models into the engine: Blinn, Cook-Torrance and Burley specular models. The BRDF components used for each model are described in Sections 4.2 - 4.4.

The only issue that occurred during the integration of specular BRDFs is the normalization of its components. In order to fix it, we added manual clamping of the dot product values before feeding them into the formulas. That way, we avoid NaN specular values.

5.2 Shaders and Materials

After these steps, the models were ready for use in Unreal Engine Editor. In order to create various materials with those models, we first created a simple shader using Material Editor¹ named Simple Lit. The shader has support for such basic properties as Base Color, Metalness, Roughness, Emissive Color and Normal map.

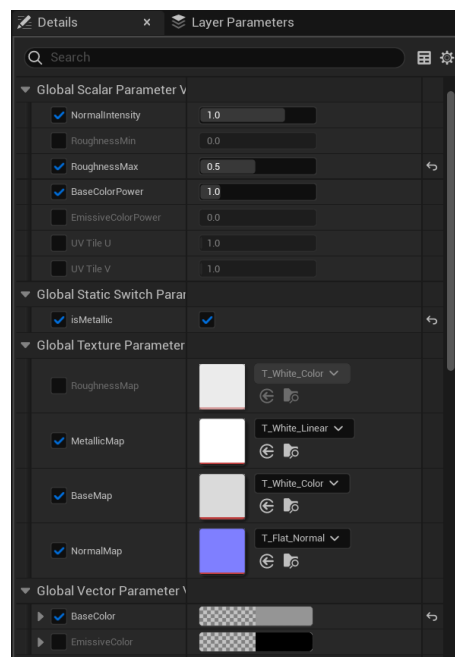


FIGURE 5.1: Properties of a material that uses Simple Lit shader.

¹<https://docs.unrealengine.com/4.27/en-US/RenderingAndGraphics/Materials/Editor/>

In order to test the models, we wanted to create a few different types of materials that simulate real-world properties. In particular, metal and plastic (rubber) materials were essential to test, as they portray unique visual properties and are usually notably reflective.

To find textures suitable for those materials, we searched the Unreal Engine Marketplace² for free materials packs. We ended up using the Twinmotion Materials pack³.

We then created three basic materials using the textures from the pack and Simple Lit shader:

- Metal material that simulates aluminum.
- Plastic (rubber) material that alongside plastic properties has a dirt mask rendered on top, so it does not appear fully smooth.
- Fully black non-metallic material with a uniform roughness value of 0.5.

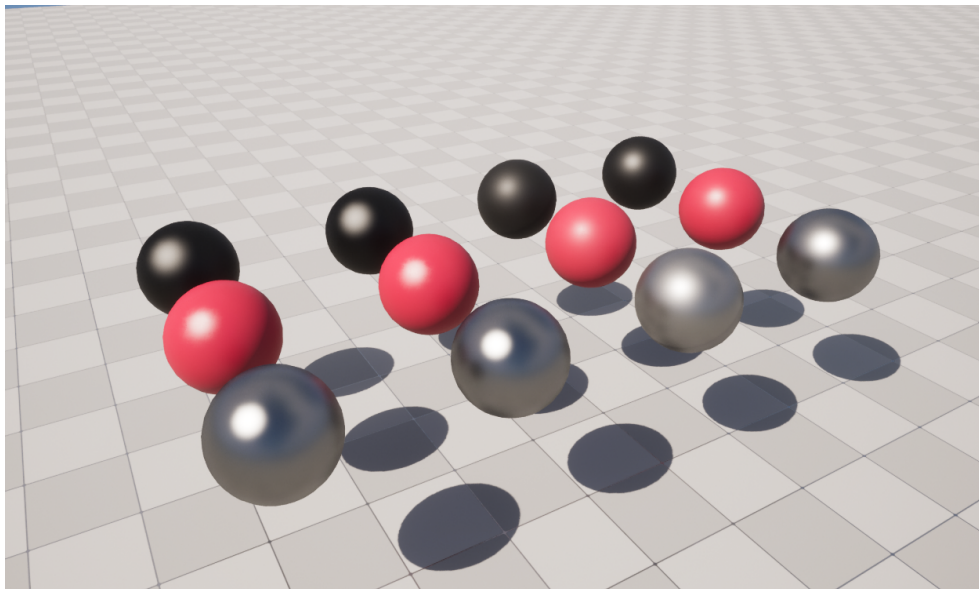


FIGURE 5.2: Rows of spheres with described above materials and different shading models. From left to right: Blinn, Cook-Torrance, Burley and Default Lit (GGX) specular model.

5.3 Visual Comparison of Models

In order to test the visual properties of each model, we first created a simple basic scene in the editor and added multiple spheres - one for each shading model and material (Fig 5.2).

We also composed a series of pictures for each material with different shading models so that they can be vividly compared side to side, as is shown in Fig 5.3 - Fig 5.5.

Although the difference between the models in the previously mentioned figures is vivid, it is not enough to portray how the use of those models will influence the

²<https://www.unrealengine.com/marketplace>

³<https://www.unrealengine.com/marketplace/en-US/product/twinmotion-materials>

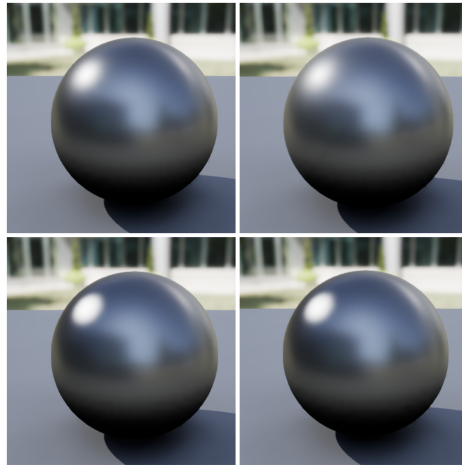


FIGURE 5.3: Metal material with different shading models. From left to right, top to bottom: Default Lit (GGX), Burley, Blinn, Cook-Torrance specular model.

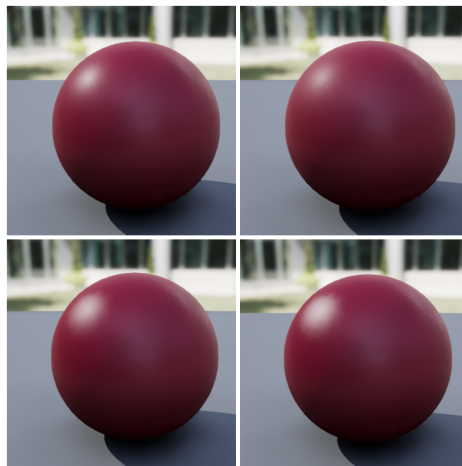


FIGURE 5.4: Rubber material with different shading models. From left to right, top to bottom: Default Lit (GGX), Burley, Blinn, Cook-Torrance specular model.

look and the feel of the actual game scenes. Due to this, we again searched the UE Marketplace for free-to-use game environments and found the Old West environment⁴, which simulates the photorealistic old town. We then added this pack to the UE project.

For the purpose of comparing visuals of different models, we found the master (parent) materials of all objects in the scene and simply switched the shading model to the desired one. We then captured the result for each model (Fig 5.6).

Because the difference between the look of this environment with different models was not very noticeable, we also tried to search the UE Marketplace for environments with more reflective objects and ended up using the Polar Sci-Fi Facility environment⁵. It simulates an abandoned facility with a number of metal-like objects. Following the same procedure as with the previous environment, we captured its look with different specular models (Fig 5.7).

⁴<https://www.unrealengine.com/marketplace/en-US/product/old-west-learning-project>

⁵<https://www.unrealengine.com/marketplace/en-US/product/polar-sci-fi-facility>

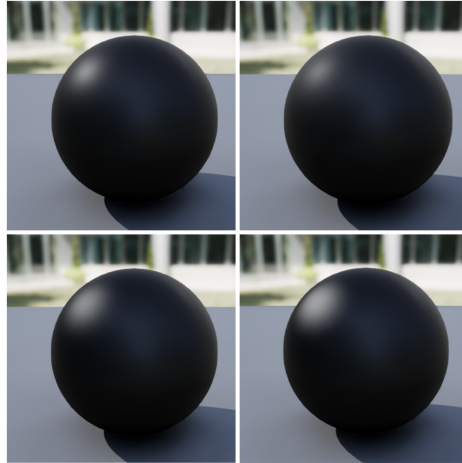


FIGURE 5.5: Solid black material with different shading models. From left to right, top to bottom: Default Lit (GGX), Burley, Blinn, Cook-Torrance specular model.

5.4 Comparison of Models by Performance Costs

Besides the visual difference, in this work, we also wanted to test the performance cost differences between the models.

One of the popular ways to test performance cost is to compare MSPF (milliseconds per frame) values. Unreal Engine has built-in tools in order to do that. There are several aspects that need to be considered for efficient profiling. Firstly, it is essential to disable the Smooth Framerate in the UE project settings. This setting snaps the value to stable numbers to stop sudden value spikes. It is also important to turn off vertical sync in the computer's graphics settings. Vertical sync, or VSync, is responsible for stabilizing the FPS value by synchronizing it with a display's refresh rate. Lastly, although testing performance in the editor is possible, it is not ideal because, unlike the game application, there are a lot of additional background processes running in the editor, so the frame rate is lower. Considering this, we packaged several versions of the same project with different shading models. We also added execution of the console command that shows FPS statistics to the level blueprint. That way, it is possible to get and analyze those values outside of the editor.

The first performance test was done on the Old West project scene. This environment is heavy on geometry, so the average MSPF value is higher. The reason for comparing the MSPF value over the FPS value is that it is more consistent than the FPS value and fluctuates not as much. Because the fluctuation is still apparent, we measured the two nearest whole numbers that the value shifts in between. The results for the Old West scene can be seen in Table 5.1.

Because this environment does not contain a lot of highly reflective surfaces, we also conducted the same test but on a scene with more vivid specular highlights. We created a scene and filled it with statue meshes from the UE starter pack. We then applied our metal material (Fig 5.3) to the statues and added some additional lights of different types. Both lights and the complex geometry of the statues increase the complexity of specular highlight calculations. We added a camera with such an angle so that it captures all of the statues and then repeated the same procedure as for the Old West environment. The results can be seen in Table 5.2.



FIGURE 5.6: Old West scene overview with different models. From left to right, top to bottom: Default Lit (GGX), Burley, Blinn, Cook-Torrance specular model.



FIGURE 5.7: Polar facility scene overview with different models. From left to right, top to bottom: Default Lit (GGX), Burley, Blinn, Cook-Torrance specular model.





| Scene Image | Model | MSPF |
|--|-------------------------|----------|
|  | Default Lit (GGX) | 27-28 ms |
|  | Burley | 27-28 ms |
|  | Blinn | 26-27 ms |
|  | Cook- Torrance | 27-28 ms |

TABLE 5.1: Milliseconds Per Frame comparison for different specular models of the Old West environment scene.

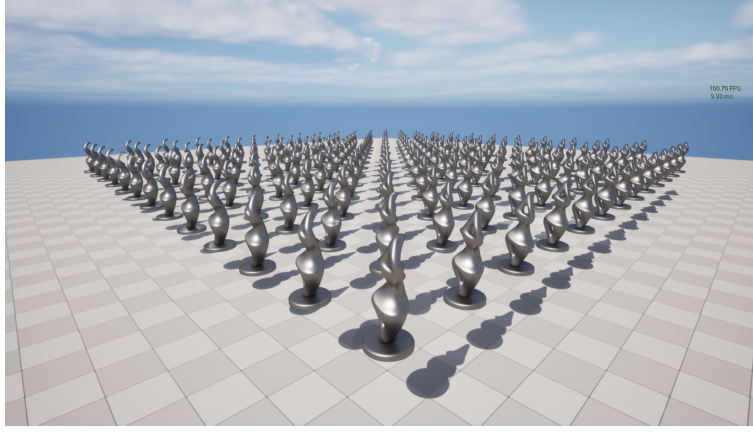
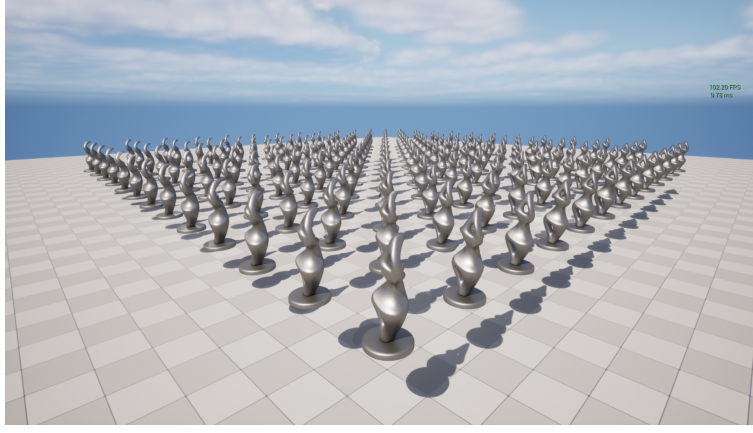
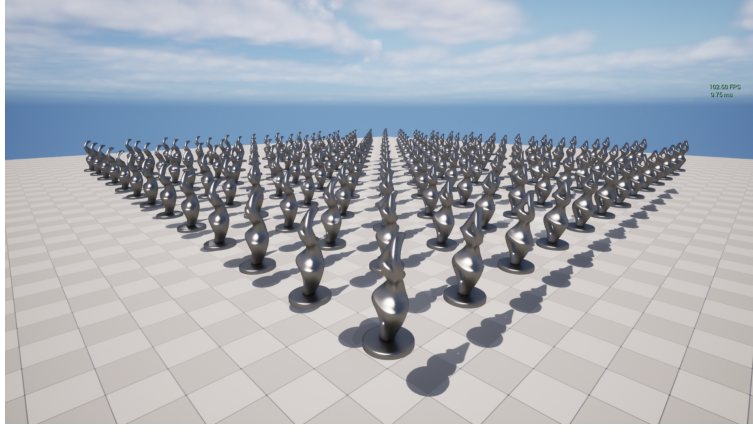
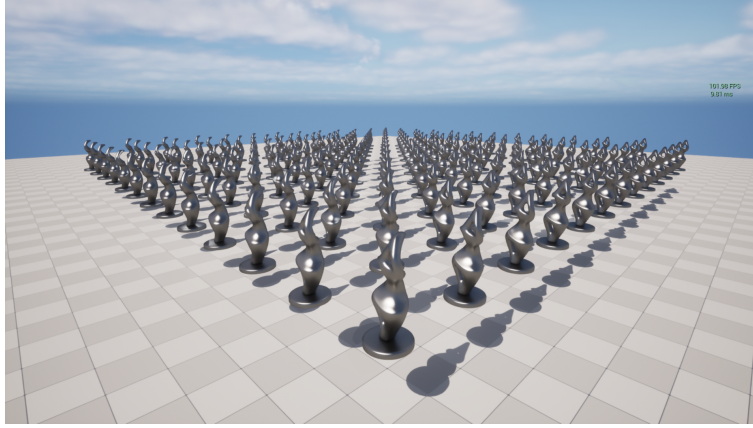
| Scene Image | Model | MSPF |
|---|-------------------|------------|
|  | Default Lit (GGX) | 9.1-9.3 ms |
|  | Burley | 9.1-9.3 ms |
|  | Blinn | 9.1-9.3 ms |
|  | Cook-Torrance | 9.1-9.3 ms |

TABLE 5.2: Milliseconds Per Frame comparison for different specular models of the scene full of metal statues.

Chapter 6

Results

6.1 Visual Difference

6.1.1 Material Comparison

The visual difference between the materials with different models, as seen in Figures 5.2 - 5.5, is very vivid. Furthermore, the difference between the Burley and GGX specular models is best seen in the metal material highlight. Burley model produces the longest tail, as it was described in the relevant article by Burley (2012). According to the same article, such a model produces results closest to real-world materials.

As for the Blinn and Cook-Torrance models, they both produce much more intense highlights than the GGX model. This is again particularly visible in metal material. The two models produce very similar results. This is because both the Beckmann distribution used in the Cook-Torrance model and the Blinn-Phong distribution are based on the Gaussian distribution function. Although the difference between the Blinn and Cook-Torrance models is very subtle, it can be noticed in plastic material (Fig 5.4) as well as in black material (Fig 5.5) - the specular highlight produced by Cook-Torrance model is slightly more intense.

6.1.2 Environment Comparison

The Old West environment scene (Fig 5.6) looks almost the same despite the specular shading model. This is most likely due to the fact that the highlights in this particular scene are mostly composed of the diffuse component. There are almost no smooth reflective surfaces, so despite the intense lighting, the difference between images is insignificant. The one image that differs from others is the scene image with Cook-Torrance shading model. The highlight here appears generally stronger. This can be mainly observed on the roof of a central building. The reason for that is likely because the geometric visibility factor in the Cook-Torrance model is based on the V-cavity approach, as discussed in Section 4.3.2. All other models use the Smith approach for geometric attenuation.

The same test conducted on the Polar facility environment was expected to produce a clearer difference due to the metal objects in the scene. However, as seen in Fig 5.7, the results for different models are again almost the same. The only change can be observed in the highlight of the metal objects located in the middle of a scene. Nevertheless, this difference is very subtle, and the general image remains almost identical.

6.2 Performance Cost Difference

The measured MSPF values that are to be observed in Tables 5.1 and 5.2 were almost the same for different models.

The Old West scene results were generally higher, as this scene is complex. However, because there are not many reflective objects, it can be assumed that specular highlights were responsible for a little part of these costs. The results measured for the Blinn model were a little better than for the rest of the models. And while it is true that, generally Blinn-Phong specular model is less expensive than the alternatives, the difference of one millisecond is very small and can be considered within error bounds. This statement especially holds if we consider the fluctuation of the MSPF value.

The scene with statues, on the other hand, shows a much better performance, therefore, smaller MSPF values. Because all objects in the scene are highly reflective and there are a lot of lights, we can also assume that specular highlights are responsible for the bigger part of the performance costs. However, the measured values for this scene were the same for every model. It can be assumed that this scene is too simple to measure a difference between specular calculations.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this work, we added three different specular shading models to Unreal Engine 5 and tested their look with different materials and game environments. We also compared the performance cost for each model.

The visual difference between the models was generally apparent, and every model showcased the expected visuals. Although the results for game environments were not as distinct as for the individual materials, the integration of such models into the games can be useful for certain game styles and materials. In this work, we provided the models that both produce softer (Burley) and more intense (Blinn, Cook-Torrance) highlights as compared to the default GGX specular model. We also managed to capture the distinction between the models that use different geometric factor approaches.

During the profiling tests, we did not manage to find a case where specular calculations considerably influenced the performance costs. It can be concluded that for the cases that were considered in this thesis, in particular, a complex scene with less reflective surfaces and a simpler scene but heavier on specular highlights, the choice of specular calculations in terms of calculations complexity can be overlooked.

7.2 Future Work

Although our version of the integrated Burley model managed to portray the desired properties and softer reflection, we hard-coded the value of the additional parameter that was described with the introduction of this model. In order for this model to be more versatile and to represent its original form closely, it can be useful to add the ability to set this parameter using the pin in a Material Editor.

We also considered improving the measurement of performance costs by setting up more complex and heavier on specular reflections scenarios. The use of different profiling methods, such as shader instruction counts, or GPU statistics analysis may also be useful to get clearer difference between the models.

Bibliography

- Ashikhmin, Michael and Peter Shirley (Jan. 2001). "An anisotropic phong BRDF model". In: *Journal of Graphics Tools* 5. DOI: [10.1080/10867651.2000.10487522](https://doi.org/10.1080/10867651.2000.10487522).
- Beckmann and Spizzichino (1963). *The Scattering of Electromagnetic Waves from Rough Surfaces*.
- Blinn, James F. (1977). "Models of Light Reflection for Computer Synthesized Pictures". In: 192–198. URL: <https://doi.org/10.1145/965141.563893>.
- Boksansky, Jakub (Feb. 2021). "Crash Course in BRDF Implementation". In.
- Burley, Brent (2012). "Physically-Based Shading at Disney". In.
- Cook, R. L. and K. E. Torrance (1982). "A Reflectance Model for Computer Graphics". In: *ACM Trans. Graph.* 1.1, 7–24. ISSN: 0730-0301. DOI: [10.1145/357290.357293](https://doi.org/10.1145/357290.357293). URL: <https://doi.org/10.1145/357290.357293>.
- Gotanda, Yoshiharu, Masaki Kawase, and Masanori Kakimoto (2015). "Real-Time Rendering of Physically Based Optical Effects in Theory and Practice". In: *ACM SIGGRAPH 2015 Courses*. SIGGRAPH '15. Los Angeles, California: Association for Computing Machinery. ISBN: 9781450336345. DOI: [10.1145/2776880.2792715](https://doi.org/10.1145/2776880.2792715). URL: <https://doi.org/10.1145/2776880.2792715>.
- Heitz, Eric (June 2014). "Understanding the Masking-Shadowing Function in Microfacet-Based BRDFs". In: *Journal of Computer Graphics Techniques (JCGT)* 3.
- Hery, Christophe and Ryusuke Villemin (2013). "Physically Based Lighting at Pixar". In.
- Hoffman, Naty (2010). "Crafting Physically Motivated Shading Models for Game Development". In.
- Karis, Brian (2013). "Real Shading in Unreal Engine 4 by". In.
- Kelemen, Csaba and László Szirmay-Kalos (Apr. 2001). "A Microfacet Based Coupled Specular-Matte BRDF Model with Importance Sampling". In: *In Eurographics Short Presentations* 25.
- Lagarde, S. and C. de Rousiers (2014). "Moving Frostbite to Physically Based Rendering". In.
- Langlands, A. (2014). "Physically Based Shader Design in Arnold". In.
- Lazarov, D. (2011). "Physically Based Lighting in Call of Duty: Black Ops". In.
- Phong, Bui Tuong (1975). "Illumination for computer generated pictures". In: *Communications of the ACM*.
- Sawicki, Dariusz (Jan. 2021). "Microfacet Distribution Function: To Change or Not to Change, That Is the Question". In: pp. 209–220. DOI: [10.5220/0010252702090220](https://doi.org/10.5220/0010252702090220).
- Schlick, Christopher M. (1994). "An Inexpensive BRDF Model for Physically-based Rendering". In: *Computer Graphics Forum* 13.
- Smith, B. (1967). "Geometrical shadowing of a random rough surface". In: *IEEE Transactions on Antennas and Propagation* 15.5, pp. 668–671. DOI: [10.1109/TAP.1967.1138991](https://doi.org/10.1109/TAP.1967.1138991).
- Snow, Ben (2010). "Terminators and Iron Men: Image-based lighting and physical shading at ILM". In: *SIGGRAPH 2010*.
- Trowbridge, T. S. and K. P. Reitz (1975). "Average irregularity representation of a rough surface for ray reflection". In: *J. Opt. Soc. Am.* 65.5, pp. 531–536. DOI: [10.1364/JOSA.65.000531](https://doi.org/10.1364/JOSA.65.000531).

- 364/JOSA.65.000531. URL: <https://opg.optica.org/abstract.cfm?URI=josa-65-5-531>.
- Walter, Bruce et al. (2007). "Microfacet Models for Refraction through Rough Surfaces". In: *Rendering Techniques*.
- Ward, Gregory J. (1992). "Measuring and Modeling Anisotropic Reflection". In: *SIG-GRAPH Comput. Graph.* 26.2, 265–272. ISSN: 0097-8930. DOI: [10.1145/142920.134078](https://doi.org/10.1145/142920.134078). URL: <https://doi.org/10.1145/142920.134078>.