Application of Unity Shader in Character Rendering: A Case Study on Dota Ogre Mag

Yunzhe Wang

Longhe Campus, Anhui University, Sanliyan Street, Shushan District, Hefei, China

r32114034@stu.ahu.cn

Abstract. The video game industry has seen remarkable growth in recent years, accompanied by an escalating demand for high-quality graphics and realistic character rendering. Unity, a leading game development engine, offers an extensive toolkit for asset creation and optimization, including shaders, which are crucial for character rendering. Despite the availability of these tools, there is a lack of comprehensive studies that delve into the practical application of Unity shaders for character rendering, particularly for popular games like Dota 2. This paper first provides an in-depth understanding of the role of shaders in Unity, focusing on their impact on the rendering process. It then presents a case study analyzing the application of Unity shaders in the rendering of the Dota 2 character 'Ogre Magi.' The study explores various shader techniques, compares their effectiveness, and discusses the challenges and solutions in achieving realistic character rendering. This experiments demonstrate that the appropriate use of Unity shaders can significantly enhance the visual fidelity of game characters, offering a more immersive gaming experience, the findings also provide valuable insights for developers aiming to optimize character rendering in Unity-based games.

Keywords: Unity, shader, character rendering, Dota 2, ogre magi.

1 Introduction

In the ever-evolving landscape of game design, Unity has carved a niche for itself as a formidable engine, facilitating the birth of both 2D and 3D games across diverse platforms. A standout feature that has garnered attention is its innovative shader development toolkit. This toolkit not only offers developers a canvas to design and tweak shaders with visual precision but also plays a pivotal role in shaping a game's visual narrative. Unity Shader, operating directly on the GPU, equips developers with a palette to conceive unique materials and lighting nuances, drawing inspiration from the Shader Language—a sophisticated programming dialect bearing similarities to Nvidia's Cg [1].

Understanding the nuances between Unity's dual shaders, the vertex and the fragment shader, is crucial. While the vertex shader molds the 3D model's vertices, the fragment shader paints each pixel's color. The visual representation of characters,

B. H. Ahmad (ed.), *Proceedings of the 2023 International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2023)*, Advances in Intelligent Systems Research 180,
https://doi.org/10.2991/978-94-6463-370-2_44

as showcased in games like Dota 2 by Valve, can be a game-changer in terms of player immersion. Dota 2, with its eclectic character ensemble, including the enigmatic Ogre Magi with its dual heads and unique traits, underscores the significance of shaders in breathing life into characters [2]. This sets the stage for an intriguing inquiry: How can we harness the full potential of Unity's shader toolkit to amplify the visual depth of such multifaceted characters?

Driven by the surging appetite for visually immersive gaming narratives and the central role of characters in weaving these tales, this study ventures into the depths of Unity's shader toolkit. By spotlighting Ogre Magi from Dota 2, this paper endeavors to unearth strategies that can redefine the visual gaming narrative. Our goal is to furnish developers with insights that can transform their game's visual resonance.

2 Rendering the Ogre: Step-by-Step Analysis

2.1 Characteristics

In Dota 2, the Ogre Magi (Fig.1) is a melee support hero known for his high base armor, decent movement speed, and strong abilities, which include a stun, a slow, and a damage buff for allies. Aesthetically, he's quite bulky, has two heads, and wears rudimentary armor. He's also known for his vibrant blue skin and carries a club as a weapon.

Highly Textured Skin: Ogre Magi's skin is not smooth; it's quite rough and wrinkled.

Dual Personality: Having two heads gives him a unique look, which often involves both heads expressing different emotions.

Armor and Club: Wears simple, rustic armor, and wields a crude club.

Color Scheme: Primarily blue skin with contrasts of brown and metal from his equipment.

Cartoonish Appearance: His features are exaggerated to make him look comical and intimidating at the same time.

Based on the above characteristics, it can analyze how to make shaders.

Fig. 1. The Ogre Magi [3]

2.2 Texture

Diffuse (or Albedo) Map: This basic color texture captures Ogre Magi's blue skin and the primary colors of his armor and club (Fig.2). It provides the base onto which other textures will add detail.

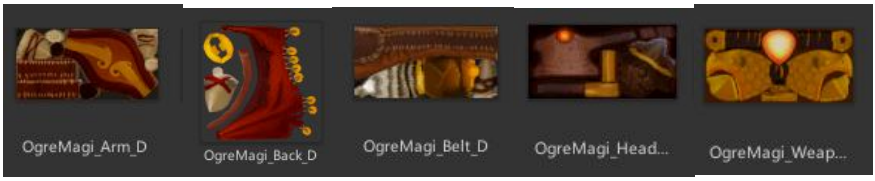

Fig. 2. Diffuse (or Albedo) Map [3]

Normal Map: Essential for adding depth and the illusion of complex surface geometry. It would accentuate the ruggedness of his skin, the grain of his club, and any intricacies on his armor (Fig.3).

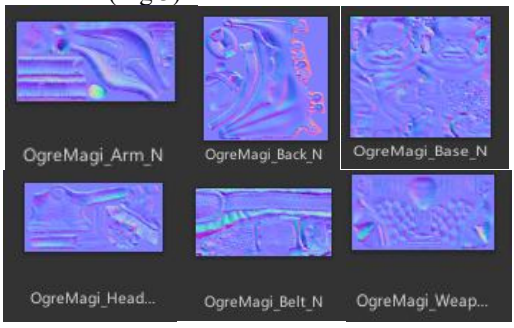

Fig. 3. Normal Map [3]

Specular Map: This map defines how the light reflects off different parts of Ogre Magi (Fig.4). His skin, being relatively matte, would have low secularity, while certain parts of his armor might reflect light more sharply.

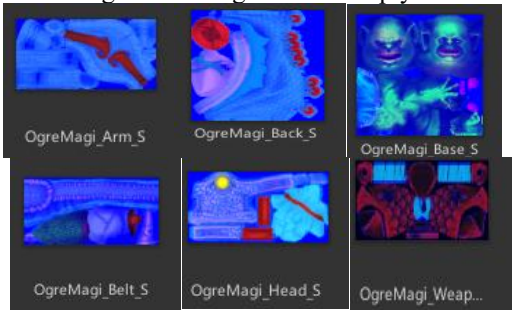

Fig. 4. Specular Map [3]

Roughness/Metallic Map (depending on the shading model): This would help differentiate between the metallic parts of his armor and the non-metallic elements like his skin and club (Fig.5).

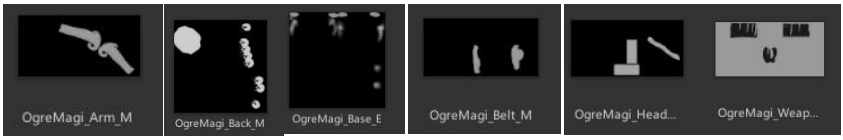

Fig. 5. Roughness/Metallic Map [3]

Emissive Map: For parts of Ogre Magi that might glow, especially when casting spells or his hands/clubs are charged with magic, an emissive map would make those parts appear self-lit (Fig.6).

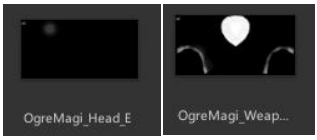

Fig. 6. Emissive Map [3]

To capture Ogre Magi's distinctive features, a combination of various texture maps should be employed in the shader creation process. These maps, when properly layered and integrated, will give depth, realism, and dynamism to his appearance in the game [4].

2.3 Lighting Models

In the process of rendering the Ogre Magi, it is essential to employ a variety of lighting models to accurately depict the character's physical attributes and maintain the artistic integrity of Dota 2. Below this paper delves deeper into the different lighting models and their specific applications for rendering the Ogre Magi [5]:

- Lambert:

Principle: This model operates on the principle of Lambertian reflectance, which assumes a perfect diffusion of light, reflecting it equally in all directions. It is ideal for surfaces that do not exhibit a glossy sheen.

Application for Ogre Magi: Leveraging Lambert shading can effectively portray the rough and matte texture of Ogre Magi's skin, offering a natural and realistic representation that aligns with the character's rugged persona.

- HalfLambert:

Principle: A variation of the Lambert model, the HalfLambert shading allows for a more gradual transition between light and shadow, reducing the contrast and ensuring shadow areas are less dark.

Application for Ogre Magi: Utilizing HalfLambert shading can retain the vibrant and somewhat exaggerated aesthetic of Dota 2 characters, preventing Ogre Magi from appearing overly dark and maintaining the cartoonish visual language of the game.

- Phong:

Principle: The Phong model introduces a specular reflection component, enabling the depiction of shiny and glossy surfaces with high reflectivity.

Application for Ogre Magi: Applying Phong shading can enhance the visual appeal of Ogre Magi's armor and potential sweat or glisten on his body during battles, providing a dynamic and realistic portrayal of reflective surfaces.

- Fresnel:

Principle: The Fresnel effect modulates the reflectivity of surfaces based on the viewing angle, increasing reflectivity as the angle between the viewer and the surface's normal augments.

Application for Ogre Magi: Implementing Fresnel shading can capture the nuanced changes in reflectivity on various surfaces, such as Ogre Magi's eyes and armor, adding a layer of realism and depth to the character's appearance.2.4Technicals

2.4 Technical Aspects

To achieve a rich and immersive representation of Ogre Magi, it is pivotal to integrate advancedtechnical aspects in the shader creation process. Here, this work explores several technicalmethodologies and their applications in enhancing the visual depiction of Ogre Magi [6]

- Cubemap

Principle: A cubemap is a collection of six square 2D textures that represent a 360-degree view of an environment. Each texture corresponds to a face of a cube, encapsulating the viewer.

Application: For Ogre Magi, cubemaps can be instrumental in simulating environmental reflections. When the character traverses different terrains or engages in battles, the reflections on his metallic armor or any shiny embellishments can dynamically change, mirroring the immediate surroundings and enhancing the player's immersion.

- FresnelSpecWarp

Principle: The FresnelSpecWarp technique is rooted in the Fresnel equations, which describe how light interacts with boundaries between two different media (Fig.7). This method accentuates the specularity based on the angle of incidence.

Application: On Ogre Magi, this can be particularly effective for surfaces like his eyes or any gem-like adornments. As the viewing angle changes, these surfaces can exhibit varying degrees of reflectivity, adding depth and dynamism to the character's appearance.

Fig. 7. FresnelSpecWarp [3]

- WarpTex

Principle: WarpTex is a procedural texture distortion technique. By using a reference texture or mathematical function, the primary texture undergoes perturbations, creating a dynamic visual effect (Fig.8).

Application: For Ogre Magi, this can simulate subtle skin movements or the flow of magical energies across his attire. Such distortions can make the character appear more lifelike, especially during intense gameplay moments.

Fig. 8. WarpTex [3]

- Rimmask

Principle: The Rimmask technique is designed to highlight the outer edges or contours of a 3D model. It often employs a gradient or mask to emphasize the model's silhouette (Fig.9).

Application: By applying this to Ogre Magi, developers can ensure that the character stands out, especially in scenes with complex backgrounds or low ambient light. This not only accentuates Ogre Magi's form but also aids players in quickly identifying the character during gameplay.

Fig. 9. Rimmask [3]

- FresnelRimWarp

Principle: This technique marries the concepts of Fresnel shading with rim effects. It dynamically adjusts the rim's intensity based on the viewer's perspective (Fig.10).

Application: When applied to Ogre Magi, this can create a pronounced halo or backlit effect, especially when the character is positioned against light sources. Such effects can underscore Ogre Magi's prominence and stature in the game's narrative.

In conclusion, the meticulous integration of these advanced methodologies not only augments Ogre Magi's visual appeal but also ensures that the character's representation is in harmony with the broader artistic and technical paradigms of Dota 2.

Fig. 10. FresnelRimWarp [3]

When creating shaders for the Ogre Magi, it's vital to utilize advanced shading techniques. These not only enhance the character's visual appeal but also ensure a consistent and realistic representation in line with the game's overall artistic direction.

3 Comprehensive Exploration of Ogre's Rendering Techniques

3.1 DiffCol (Diffuse Color)

At the heart of character rendering lies the DiffCol technique. This method is not just about assigning a color to an object; it's about understanding how an object's inherent color interacts with a neutral white light source. Within the context of Ogre Magi, the DiffCol technique plays a pivotal role in defining foundational attributes, such as the character's skin tone and attire. Unity's shader algorithm, in its intricate design, computes the interaction between the light vector and the surface's normal vector. This interaction, when multiplied with the texture's inherent color values, results in a nuanced representation of the object's color under specific lighting conditions [7]

3.2 SpecCol (Specular Color)

The SpecCol technique introduces a layer of sophistication to character rendering. It seeks to emulate the intricate ways in which light reflects off glossy surfaces, producing highlights that our eyes associate with shininess or gloss. Unity's standard shader, in its implementation, leverages the Phong reflection model. This model, celebrated for its balance between computational efficiency and visual realism, captures the nuanced reflections based on the viewer's position relative to the light source and the object.

3.3 Fresnel

The Fresnel effect, a cornerstone in rendering realism, is named after the physicist Augustin-Jean Fresnel. It's a technique that captures the unique behavior of light as it transitions between different media. Unity's shader intricately employs the Fresnel-Schlick approximation, which is revered for its computational efficiency. This approximation ensures that objects appear more reflective at grazing angles, a phenomenon commonly observed in real-world scenarios [8].

3.4 DirDiff (Directional Diffuse)

The DirDiff technique is a testament to the complexities of light behavior. It focuses on how light, upon striking a surface, scatters directionally. Unity's approach is rooted in the Lambertian reflectance model, a time-tested model that assumes light scatters uniformly in all directions. This model is particularly effective for surfaces that exhibit a matte finish, ensuring that the character's form is illuminated in a realistic manner, especially under direct sunlight.

3.5 DirSpec (Directional Specular)

Building upon the foundational principles of specular reflection, the DirSpec technique introduces an element of directionality. This ensures that the reflections on the character's armor and weaponry are not just based on the viewer's position but also on the orientation and directionality of the light source. Unity's implementation seamlessly integrates the Phong reflection model with additional parameters that account for the light's orientation, enhancing the reflection's realism and depth.

3.6 EnvDiff (Environmental Diffuse)

Ambient lighting, often overlooked, plays a crucial role in setting the mood and tone of a scene. The EnvDiff technique simulates this ambient light, ensuring that characters like Ogre Magi are rendered realistically in indoor or dimly lit settings. Unity employs a sophisticated technique known as Image-Based Lighting (IBL). IBL, in its essence, uses a precomputed texture that encapsulates the environment's lighting nuances, providing indirect lighting information that adds depth and realism to the scene.

3.7 EnvSpec (Environmental Specular)

The EnvSpec technique is a masterclass in rendering reflections. It focuses on how glossy surfaces, like armor or enchanted artifacts, reflect their environment. Unity's approach is a blend of pre-filtered environment maps and the principles of Physically-Based Rendering (PBR). PBR, in its design, considers the microscopic imperfections and details of a surface to produce reflections that are incredibly realistic. The

interplay of light and shadow, combined with the environment's reflections, ensures that characters like Ogre Magi stand out, enhancing the overall gaming experience.

In conclusion, the meticulous integration of these advanced rendering techniques ensures that characters are not just visually appealing but are rooted in realism. The attention to detail, combined with the computational prowess of platforms like Unity, paves the way for an immersive and visually captivating gaming experience.

4 Rendering Results and Analysis

4.1 DiffCol and SpecCol Integration

The crux of our visualization methodology hinges on the seamless fusion of DiffCol and SpecCol. DiffCol, reminiscent of the Diffuse map, is shaped by the intricate interplay of the Color texture, Metalness, and TintByBase indicators. To put it differently, DiffCol is crafted by harmonizing the foundational color with an absolute black hue, contingent on its metallic attributes.

Simultaneously, the SpecCol, evocative of the SpecularColor map, is ascertained. This is realized by melding the foundational hue with a distinct gray tint (established empirically as 0.3) and subsequently amplifying the outcome with the specInt parameter (Fig.11).

The hues derived from these intricate computations lay the groundwork for the advanced visualization phases that follow.

Fig. 11. DiffCol and SpecCol Integration (Photo/Picture credit: Original)

4.2 Fresnel Adjustments

The Fresnel effect is then incorporated, taking into account the unique characteristics of both metallic and non-metallic surfaces. A Lerp operation is performed on the Fresnel values to attenuate the metallic component (Fig.12). The Fresnel values are then extracted for various channels:

R: FresnelCol, which is an interpolation mask of two DiffWarps. However, its use is rare and thus not considered in this context.

G: FresnelRim, which is used for rim lighting.

B: FresnelSpec, employed for specular reflection.

Fig. 12. Fresnel Adjustments (Photo/Picture credit: Original)

4.3 DirDiff: Main Light Diffuse Reflection

The main light diffuse reflection is enhanced by introducing a primary light color, termed LightCol. This is followed by the declaration of corresponding input parameters. The main light diffuse reflection utilizes RampTex through the following steps:

Calculation of HalfLambert: HalfLambert calculation modifies Lambertian reflectance for better shading in low light.

Construction of RampUV:RampUV construction maps a ramp texture onto the object's surface, utilizing lighting information for nuanced shading.

Sampling of RampTex:RampTex sampling retrieves shading values from the texture using RampUV coordinates

Blending of DiffCol, Ramp results, and primary light color:blending combines the diffuse color (DiffCol), Ramp Texture results, and primary light color (LightCol) to achieve realistic shading on the surface, ensuring a visually appealing and realistic interaction of light with the material [9], enriching the rendering's visual realism and depth (Fig.13).

Fig. 13. Main Light Diffuse Reflection (Photo/Picture credit: Original)

4.4 DirSpec: Main Light Specular Reflection

The DirSpec technique accentuates the main light specular reflection, enhancing the visual depth and realism of objects under primary light sources. This technique introduces new parameters to the material panel, including SpecPow and _Specnt, which govern the specular exponent and intensity, respectively. The process employs a combination of Phong and FresnelSpec methodologies, facilitating a nuanced

approach to specular reflection, where the interplay of light and material properties is depicted with a high degree of realism (Fig.14).

Fig. 14. Main Light Specular Reflection (Photo/Picture credit: Original)

4.5 EnvDiff: Environmental Diffuse Reflection

Transitioning to the environmental aspects, the EnvDiff technique focuses on environmental diffuse reflection, a vital component in achieving a balanced and realistic lighting environment. This technique introduces EnvCol and EnvDiffint to the material panel, which dictate the environmental light color and diffuse reflection intensity, respectively. The process employs a monochromatic environmental light approach, where the DiffCol is seamlessly blended with EnvCol and EnvDiffint, resulting in a visually harmonious representation (Fig.15).

Fig. 15. Environmental Diffuse Reflection (Photo/Picture credit: Original)

4.6 EnvSpec: Environmental Specular Reflection

The EnvSpec technique delves into the realm of environmental specular reflection, a sophisticated process that enhances the visual realism of objects within an environmental context. This technique augments the material panel with EnvSpeclnt, which governs the environmental specular reflection intensity [10]. The process

employs a Cubemap methodology, facilitating a nuanced approach to specular reflection, where the interplay of light and material properties is depicted with a high degree of realism (Fig 16).

Fig. 16. Environmental Specular Reflection (Photo/Picture credit: Original)

4.7 RimLight: Rim Lighting

The RimLight technique focuses on rim lighting, a vital component in enhancing the visual depth and contour of objects. This technique introduces RimCol and Rimlnt to the material panel, which dictate the rim light color and intensity, respectively. The process employs a monochromatic environmental light approach, where the RimCol is seamlessly blended with FresnelRim and the G-channel of the normal, culminating in a visually captivating representation (Fig. 17).

Fig. 17 Rim Lighting (Photo/Picture credit: Original)

4.8 Emission: Self-Illumination

The final phase in the rendering pipeline involves the emission technique, which focuses on self-illumination, a critical component in achieving a visually immersive representation. This technique introduces _Emitlnt to the material panel, which governs the self-illumination intensity [11]. The process employs a monochromatic

environmental light approach, where the DiffCol is seamlessly blended with Emitlnt, resulting in a visually harmonious representation (Fig.18).

Fig.18 Self-Illumination (Photo/Picture credit: Original)

4.9 Final result

The rendering methodology delineated in Section 4 manifests a sophisticated interplay of techniques, meticulously orchestrated to engender a representation that is both visually compelling and grounded in realism, and make the ogre rendered more visually appealing (Fig.19).

Fig. 19 Final result (Photo/Picture credit: Original)

5 Conclusion

This research paper embarked on a comprehensive exploration into the nuanced application of Unity shaders, particularly emphasizing their role in character rendering. Using the iconic 'Ogre Magi' from Dota 2 as a case study, the research illuminated the multifaceted processes and techniques that underpin the art and science of character rendering in modern gaming.

At the outset, the paper demystified the foundational principles of shaders within the Unity framework, highlighting their indispensable role in the rendering pipeline. Through a meticulous step-by-step analysis, the research journeyed from the basics of

texture mapping to the intricacies of advanced lighting models and the technicalities of shader programming. Each phase of the rendering process was dissected, revealing the depth of detail and precision required to bring a character like Ogre Magi to life.

The results of this study were unequivocal. When harnessed effectively, Unity shaders can dramatically enhance the visual fidelity of game characters. The rendering methodologies applied to Ogre Magi not only accentuated his visual appeal but also enriched the overall gaming experience, offering players a more immersive and realistic engagement with the character. This research, therefore, stands as a testament to the transformative power of Unity shaders and offers invaluable insights for developers and artists striving for excellence in Unity-based game development.

Peering into the future, the gaming industry's trajectory suggests an ever-growing appetite for heightened realism and immersion. As such, there is an impending need for even more sophisticated rendering techniques. Future research avenues might delve into the integration of artificial intelligence (AI) in refining the rendering process, exploring adaptive shaders that respond in real-time to in-game environments, or investigating the potential of Unity shaders within the burgeoning realms of virtual reality (VR) and augment

References

1. Haas J K. A history of the unity game engine. Diss. Worcester Polytechnic Institute, 2014, 483(2014): 484.
2. Va H, Choi M H, Hong M. Real-time cloth simulation using compute shader in Unity3D for AR/VR contents. Applied Sciences, 2021, 11(17): 8255.
3. https://help.steampowered.com/zh-cn/faqs/view/299C-D7F9-09A5-98B6
4. Chiu Y P, Sun H, Liu K T. Study on Applying Brightness Fluoroscopy on Strengthening Art Appearance for 3D Game. ICIC express letters. Part B, Applications: an international journal of research and surveys, 2016, 7(6): 1313-1321.
5. Munkberg J, Hasselgren J, Shen T, et al. Extracting triangular 3d models, materials, and lighting from images, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 8280-8290.
6. McCool M D, Heidrich W. Texture shaders, Proceedings of the ACM Siggraph/Eurographics workshop on Graphics hardware. 1999: 117-126.
7. He Y, Foley T, Hofstee T, et al. Shader components: modular and high performance shader development. ACM Transactions on Graphics (TOG), 2017, 36(4): 1-11.
8. Karis B, Games E. Real shading in unreal engine 4. Proc. Physically Based Shading Theory Practice, 2013, 4(3): 1.
9. Maughan C, Wloka M. Vertex shader introduction. NVIDIA Technical Brief, 2001.
10. Šmíd A. Comparison of unity and unreal engine. Czech Technical University in Prague, 2017: 41-61.
11. Labschütz M, Krösl K, Aquino M, et al. Content creation for a 3D game with Maya and Unity 3D. Institute of Computer Graphics and Algorithms, Vienna University of Technology, 2011, 6: 124.