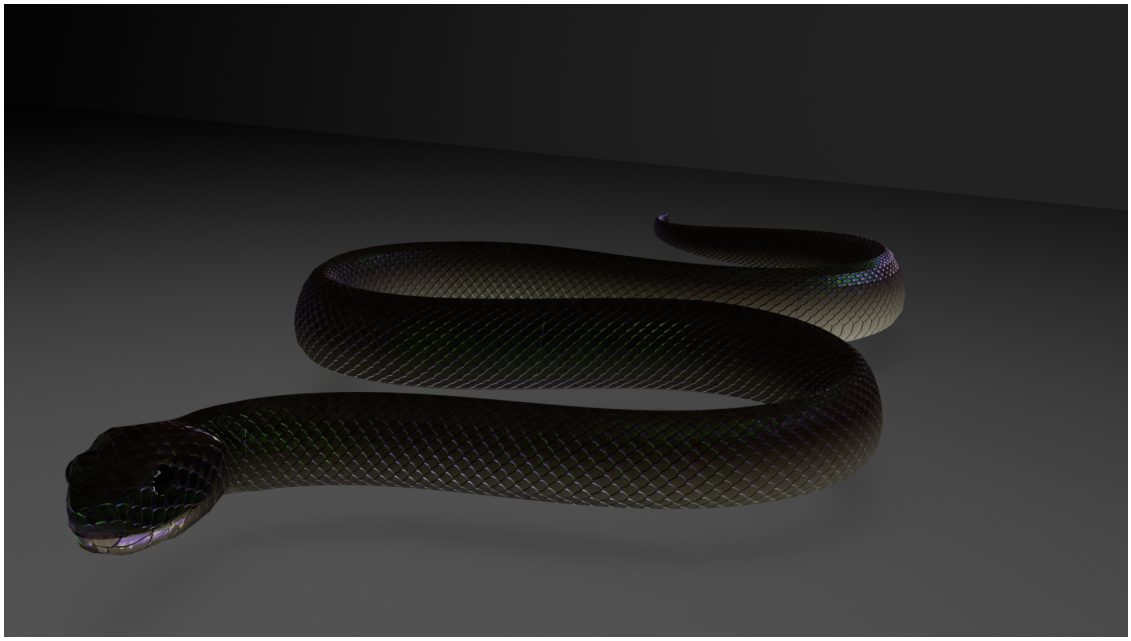


Real-time physically based snake scale rendering

Master's thesis in Computer science and engineering

Yuanshan Bie
Mingxiao Jiang

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

Real-time physically based snake scale rendering

Yuanshan Bie
Mingxiao Jiang

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Real-time physically based snake scale rendering

Yuanshan Bie
Mingxiao Jiang

Supervisor: Erik Sintorn, Department of Computer Science and Engineering
Examiner: Ulf Assarsson, Department of Computer Science and Engineering

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Path tracing rendering result of white-lipped python

Typeset in LaTeX
Gothenburg, Sweden 2024

Real-time physically based snake scale rendering

Yuanshan Bie
Mingxiao Jiang
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

There are thousands of types of snakes in the world, and many of them have a unique snake skin appearance caused by the complex structure on their surface. To create and render physically realistic snake skins in application in real time, we need to efficiently and accurately express the visual effects that snake skins should have.

In our work, we model the complex multi-layer structure of the snake skin surface as a multi-layer BRDF and integrate it into a physically based rendering model. We show the results of using this lighting model to render snakes under different parameters and configurations, and show that our model can match the appearance of real-life snake skin to a certain extent. We also implement a procedural texture generation model that can synthesize various textured skin patterns of snakes and efficiently provide the special texture required for our proposed model. The generator uses image processing to generate textures. The results show good visual similarity to the real snake skins.

Keywords: Snake skin, Multi-layer BRDF, Procedural texture generation, Rendering.

Acknowledgements

We want to thank our supervisor, Erik Sintorn, for his support throughout this project. During the start of the project, he helped us to better determine the direction of our research. During the implementation and experiment, he gave us many suggestions on model design and debugging. During the writing, he helped us review and revise the report, which greatly improved the quality of the paper. We also would like to thank all providers of those beautiful snake photos in this paper. Without them to share the photos with other people, we wouldn't have thought of this interesting direction to work in.

Yuanshan Bie & Mingxiao Jiang, Gothenburg, 2024-06-12

Contents

List of Figures

List of Tables

1

Introduction

1.1 Background

There are many snake species and they can be found on every continent except Antarctica and some islands [1]. From ancient times to the present, snakes appear in many artworks as a common cultural symbol. In recent years, with the development of virtual reality and video games, to enrich players' experience in games, or to provide a new teaching method for schools or museums, more and more applications try to depict snakes in digital media. This means creating 3D models of snakes, texturing them, and rendering them using some kind of shading model. To improve the immersion of games and for teaching purposes, the shading model needs to render the models realistically.

Physically based rendering (PBR) is a common shading method used in applications that require rendering models realistically. It is used to render models according to a mathematical model of the interaction between light and the model surface. As a basic functionality, PBR is internally supported in many renderers and game engines. For common simple object surfaces, such as metal, plastic, wood, etc., the basic PBR model can easily simulate their interaction with light and obtain realistic rendering results by slightly modifying the parameters. But for relatively complex surfaces, such as the constituent elements on the biological epidermis (such as feathers, hairs and scales), their structural complexity at multiple scales leads to complex light-matter interactions. It is difficult for the basic PBR model to simulate the properties of its surface, the model needs to be modified in conjunction with the relevant optical mechanisms.

Snake skin has a relatively complex layered structure, the surface is a combination of micro structure and pigmentation that manifests itself in interesting appearances on a macro scale, such as iridescent reflections on its surface, as shown in Figure 1.1. Since snakes currently appear less frequently in movies, animations and games than other creatures, most of the current common PBR shading models in the industry ignore the complex structure of snake skin. Generally speaking, the appearance designs of snake characters in commercial works are modeled and set up by skilled artists using time-consuming appearance matching. This increases the cost and limits its applicability in the field. For example, in the field of predictive rendering in paleontology [3] or teaching purposes, physically based rendering of models is necessary. How to perform physically based real-time rendering of snake skin is still

Figure 1.1: Photo of Northern White-lipped Python [2] , notice the iridescent reflection on its skin.

a problem to be solved.

1.2 Research Question

Given the importance of rendering snake skin realistically in real-time, we will try to solve the following research question: How to build a physically based shading model that can simulate the appearance of snake skin in reality in real-time?

1.3 Our work

Our main research work is as follows:

1. We model the snake skin surface as a multi-layer surface based on the work by Griffe [4]. Building upon the multi-layer Bidirectional Reflectance Distribution Function (BRDF) simplification approach suggested by Weidlich and Wilkie [5], we derive our own multi-layer BRDF model for snake skin rendering and integrate it into the physically based rendering model. Within this model, the uppermost layer is a thin film, engineered to replicate the iridescent effects attributable to thin-film interference. The intermediate layer serves as an absorption medium, mimicking the light absorption properties of pigment cells found in snake skin. The bottom layer is designed for diffuse reflection, emulating the light reflection characteristics of the snake skins base. Utilizing this shading model, we have rendered snakes under different settings, demonstrating that our model can approximate the appearance of real-world snake skin to a notable degree.

2. We introduce a novel procedural texture model, which is an extension of the concepts presented by Pinheiro [6]. Our model is capable of generating a diverse array of textured skin patterns akin to those found in snakes. It further accommodates the creation of different scale types and efficiently produces the specialized maps requisite for our snake skin shading model. This model is informed by the visual representations of a substantial number of real-world snakes and employs image processing techniques to synthesize textures. Our findings indicate a high degree of visual congruence with the actual skin of snakes. The textures generated by our model are versatile, finding applications not only in computer graphics but also serving educational purposes by elucidating snakes and their distinctive visual characteristics.

1.4 Structure of the paper

The paper is divided into five chapters. The main contents of each chapter are briefly described as follows:

- Chapter 1 is an introduction that discusses the research background, explains the work of this paper, and arranges the content of all chapters.
- Chapter 2 briefly introduces current developments in the field of snake skin rendering, multi-layer material, and procedural texture generation.
- Chapter 3 introduces how to model the snake skin, the basic theory of PBR, the procedural generation methods, as well as the principle of thin film interference.
- Chapter 4 introduces the procedural textures generation process of the textures used by the shading model, and then introduces the implementation and rendering process of the multi-layer BRDF model in detail.
- Chapter 5 first shows various snake skin textures produced by the procedural generation process, and then tests the shading models with different parameters and configurations to compare their render results.
- Chapter 6 summarizes the work done in this paper, points out its shortcomings, and discusses future research and development directions.

2

Related Work

The complexity of the appearance of real-world snake skin scales is due in part to its underlying structure and the optical mechanisms involved in its coloration and scattering, as well as naturally occurring regular and irregular patterning. In this section, we review the most relevant works related to our shading model and our procedural texture generation method.

2.1 Snake skin shading methods

In the industry, there are many works on physically based shading models of biological surface elements. Due to the needs, most work focuses on human skin [7][8][9] and mammal hair [10][11][12]. The works are all based on modeling and do reasonable abstraction to surface elements (such as the cross-sections of skin and hair structures) mathematically, and finally obtain a usable shading model.

For snake skin, according to research by herpetologists Shawkey and D'Alba [13], the complexity of its appearance is due to the layered structure of its surface and the arrangement of cells on the microscale level, which results in a wave phenomenon when light propagates on its surface, that is, light interference and diffraction. In this area, Smits and Meyer [14] first proposed a model accounting for thin film interference. The authors review the laws of physics that govern thin film interference and simulate the colors in nature that are produced by this mechanism. They reproduce common examples of thin film interference, including the iridescent colors created by multiple film layers. Then, Dhillon et al. [15]proposes a method for modeling snake skin accounting for the diffraction that comes from nano-structures in the scales. The approach involves using atomic force microscopy (AFM) to capture accurate surface geometries of these nanostructures, which are responsible for structural coloration due to wave interference. The method generates bidirectional reflection distribution functions (BRDFs) directly from the measured data and employs pre-computation of lookup tables to enable real-time performance. However, their model requires capturing the surface nano-structure and performing a Fourier transform of the resulting height field, making the approach expensive and difficult to generalize. In our real-time snake skin shading model, we use thin film interference to simulate the iridescent appearance of snake skin.

In addition to the wave optics, the layered structure of snake skin also has a great influence on its appearance. In terms of multi-layer materials, Hanrahan and

Kruger [16] first introduced an analytical single-scattering model for layered materials. Later, Weidlich and Wilkie [5] proposed a method to approximate multilayer materials, which ignores multi-scattering and is based on a variety of reasonable assumptions. It uses one BRDF for each layer, and combines multiple BRDFs to approximate the final multi-layer material. The latest is the method proposed by Guo et al. [17], which exploits the position-free nature of BSDF and simulates the transmission of light between multilayer materials by using bidirectional Monte Carlo path tracing with multiple importance sampling. Recently, Griffe et al. [4] first used multi-layer BSDF to approximate the layered structure of the snake skin surface and achieved good rendering results. However, their approach to building multi-layer BSDF requires using Monte Carlo path tracing, which takes several minutes to render a single image and can not be used in real-time applications. In our real-time snake skin shading model, we use the method proposed by Weidlich [5] to approximate the light propagation process and compute simulations of light transmission in multi-layer materials.

2.2 Procedural snake skin texture generation

Rendering the snake skin requires different kinds of maps, including diffuse color map for the skin, normal map, height maps, iridescent layer thickness map, etc. These maps are difficult to obtain by scanning specimens or hand-drawing, but because of the regularity of the snake skin scales, they can be generated easily using procedural texture generation methods.

This area was first studied by Cocho, Pérez-Pascual, and Rius [18]. They used cellular automata for random texture generation, but the resolution was too low to be used in current applications. Recently, Pinheiro and Walter [6] proposed a complete method for generating snake skin textures. It uses geometric calculations, cellular automata and noise to generate different snake skin textures, and change the appearance of results easily by adjustment parameters, but the generated scales only support one type of shape and do not support the generation of special maps such as iridescent layer thickness maps required by our proposed shading model. In our work, we use geometric calculations and noise to generate all the maps needed for the shading model.

3

Background theory

3.1 Snake skin color and structure

In this work, we are interested in the skin of snakes. We need to know how its epidermal structure and material properties influence its appearance in the real world. Below we first describe the snake skin on the macro scale, which mainly is the pattern of the scales. Then, for physically based rendering, we describe its inner structure and introduce a simplified snake skin model for rendering, and finally we briefly introduce thin-film interference, which we will use in our work.

3.1.1 Macro scale appearance

When examining snake skin at the macro scale, the standout feature is undoubtedly the diversity of scale patterns. These patterns vary widely among snake species. Some snakes display a uniform color throughout, while others exhibit intricate patterns that incorporate multiple hues. The complexity of these patterns can range from straightforward single stripes to elaborate designs that defy simple descriptions. In the work referenced by Jefferson [6], the classification of ophidian integumentary designs is systematically delineated into the subsequent categories.

Figure 3.1: Left (Solid color): Philodryas aestiva [19]. Mid (Vertical stripe): Phalotris lemniscatus [20]. Right (Horizontal stripe): Micrurus diastema [21].

1. Solid color. The snake does not have any pattern and is composed of only a single solid color.

2. Vertical stripe. The snake has one or multiple stripes on its body, running from its neck to its tail, or from its head to its tail.

3. Horizontal stripe. The snake has one or multiple stripes arranged perpendicular to the axis of their bodies, like rings. The rings can have one or multiple colors.

4. Spot. The snake has multiple spots on its body. The spots can have different colors. The shape of the spots can be round, ring, or other irregular ones.

5. Other simple patterns. The snake has one or multiple irregular shape patterns along the body.

6. Complex pattern. When the pattern is none of the above, it is categorized as Complex. Typically, this includes intricate (or abstract) camouflage patterns, that would be hard to describe verbally, or with numerous different elements on its body [6].

Figure 3.2: Left (Spot pattern): Ptychophis flavovirgatus [22]. Mid (Simple patterns): Bothrops alternatus [23] Right (complex pattern): Python regius [24] .

Besides the pattern, different snakes can also have different shape scales. According to Pinheiro and Walter [6], most snake scales are leaf-shaped. Some scales are smooth and others may have a protruding ridge along its longitudinal direction. Those are called keeled scales, an evolution of smooth scales seen in some snakes that cause light to disperse through the surface, which increases the roughness of the skin surface.

3.1.2 Micro scale structure

At the micro scale level, according to previous research [4] on the snake skin model, the snake skin can be divided into multiple parallel layers. The most relevant layer to appearance is the top layer. This layer can also be divided into multiple sub-layers.

At the outermost sub-layer of the top layer, we find the iridescent layer. This layer consists of structures made up of small flakes that interact with light as delicate thin-film elements. Depending on whether these flakes are organized in the same orientation or not, they will produce varying degrees of visible iridescence. Beneath the iridescent layer, we encounter a dark pigment layer, primarily composed of eumelanin. In this layer, disordered melanocytes absorb light, contributing to the overall coloration of the snake skin. Finally, at the bottom lies the soft tissue base, which reflects light in a diffuse manner.

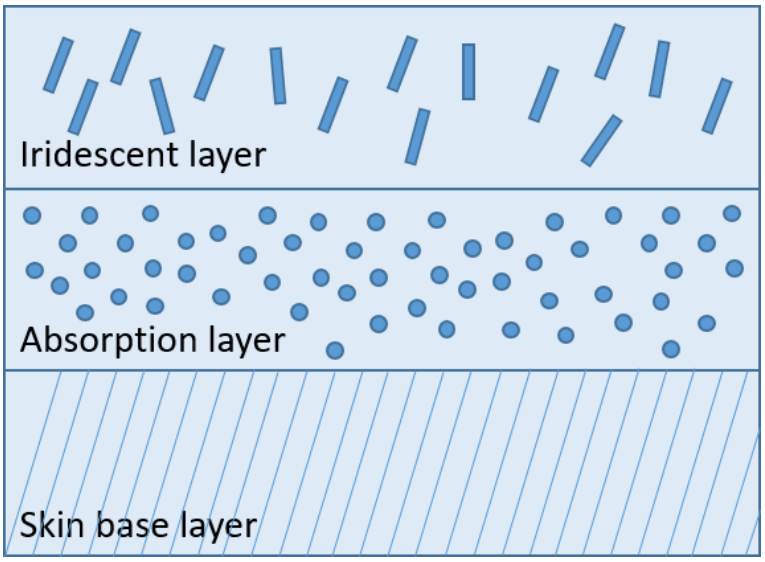


Figure 3.3: The layer structure of the snake skin top layer

3.2 Physically based rendering

Physically Based Rendering (PBR) is a method of rendering that emulates the way light is reflected in reality. It follows the rules of the interactions between light and objects. The theory of PBR is detailed in the book by Pharr, Jakob, and Humphreys [25]. This section introduces some fundamental concepts of PBR, including the rendering equation, the Bidirectional Reflectance Distribution Function (BRDF), and its extension model, the multi-layer BRDF.

The rendering equation is a core concept in computer graphics that describes how light interacts with surfaces to produce their visible appearance. It is an integral equation based on the law of conservation of energy, which states that the light leaving a point in space is the sum of all the light emitted and reflected from that point. The equation is described as:

$$L_o\left(\mathbf{x}, \omega_\mathrm{o}, \lambda, t\right) = L_\mathrm{e}\left(\mathbf{x}, \omega_\mathrm{o}, \lambda, t\right) + L_\mathrm{r}\left(\mathbf{x}, \omega_\mathrm{o}, \lambda, t\right) \tag{3.1}$$

The reflectance equation is a restricted special case of the full rendering equation, presented by Kajiya in 1986 [26].

$$L_o(p, \omega_o) = L_e(p, \omega_o) + \int_\Omega f_r(p, \omega_i, \omega_o) L_i(p, \omega_i)(\omega_i \cdot n) d\omega_i \tag{3.2}$$

In (3.2) L_o is the outgoing radiance at point p in direction ω_o. L_e is the emitted radiance from the point. $\int_\Omega$ represents the integral over the hemisphere Ω around point p. f_r is the bidirectional reflectance distribution function (BRDF), which defines how light is reflected at an opaque surface. L_i is the incoming radiance to point p from direction ω_i. $\omega_i \cdot n$ is the cosine of the angle between the incoming light direction and the surface normal n, accounting for the angle of incidence.

The equation integrates all incoming light, modified by the BRDF and the angle of incidence, to determine the outgoing light. Solving this equation for complex scenes is the primary challenge in realistic rendering.

3.2.1 Microfacet BRDF

The Bidirectional Reflectance Distribution Function (BRDF) is a critical component of the rendering equation. It describes how light reflects off surfaces and plays a key role in creating realistic images. It can be roughly classified into models based on experience, and models based on physical analysis. We mainly focus on models based on physical analysis.

In the real world, most object surfaces are irregular, causing incident light to scatter randomly. The microfacet theory posits that surfaces are composed of numerous microfacets, each with varying orientations. When incident light interacts with the surface, it splits into reflected and refracted light, with each light ray bouncing only once between surfaces. Multiple bounces are considered negligible. Consequently, this theory interprets the random distribution of light scattered by a rough surface as essentially a random distribution of microfacet orientations.

Models based on physical analysis incorporate effects such as microfacet reflection, enabling a more accurate representation of the relationship between incident and reflected light. We will briefly introduce a physically-based diffuse reflection model (Lambertian) and a widely used physically-based model (Cook-Torrance).

The Lambertian model is a fundamental diffuse model that does not account for specular reflection. It assumes all incident light enters the object, scatters internally, and is then reflected uniformly in all directions on the hemisphere. For isotropic objects, the rendering result should be consistent from all directions, meaning the viewing angle does not affect the reflection.

$$f_{Lambertian} = \frac{c_d}{\pi} \tag{3.3}$$

(3.3) shows the formula of the Lambertian BRDF, c_d is the measure of diffuse reflection. In the physically based rendering process, energy conservation is very important. The incident energy E_i should be greater than or equal to the outgoing energy E_o. The Lambertian model assumes that the outgoing light is evenly distributed over the upper hemisphere. According to spherical integration, we need to divide the result by π to ensure energy conservation.

The Cook-Torrance model is widely used today and is modeled based on micro-facet theory. Compared to the Lambertian model, it models the highlight reflection of the surface:

$$f_{total} = K_d \times f_{Lambertian} + K_s \times f_{Cook-Torrance} \tag{3.4}$$

$$f_{Cook-Torrance}(\omega_i, \omega_o) = \frac{D(h)F(\omega_i)G(\omega_o)}{4(\omega_i \cdot n)(\omega_o \cdot n)} \tag{3.5}$$

It can be seen from (3.4) and (3.5) that compared to the Lambertian model, the Cook-Torrannce model is also affected by three terms: D, F and G, which are the normal distribution term, Fresnel term, and geometric term respectively.

D is the normal distribution function (NDF) used to estimate microfacets. It can calculate the number of microfacets that are consistent with the orientation of the halfway vector(h) and also considers the influence of surface roughness factor α. GGX [27] is one of the commonly used normal distribution functions, which is calculated as follows:

$$D(h) = \frac{\alpha^2}{\pi((h \cdot n)^2(\alpha^2 - 1) + 1)^2} \tag{3.6}$$

In (3.6), h is the halfway vector, α is the surface roughness factor which is calculated by:

$$\alpha = roughness^2 \tag{3.7}$$

F is the estimated Fresnel reflection term. It can simulate the effect that when the angle between the observer and the surface changes, the corresponding reflectivity of the surface will change. Schlick's approximation [28] is one of the commonly used Fresnel term approximations, which is calculated as follows:

$$F(\theta) = F_0 + (1 - F_0)(1 - cos\theta)^5 \tag{3.8}$$

where

$$F_0 = (\frac{n_1 - n_2}{n_1 + n_2})^2 \tag{3.9}$$

In (3.8), θ is the angle between incident light direction and surface normal. $F0$ is the reflection coefficient. In (3.9), n_1 and n_2 are the indices of refraction of the two media at the interface.

G is the geometric term, which tells how much the reflected light is blocked by other adjacent micro-facets due to surface roughness. There are many different ways to calculate the G term. One of the approximations based on the Smith Joint Masking-Shadowing Function citeEric14 is calculated as follows:

$$G(l, v, h) = \frac{0.5}{G_1(l) + G_1(v)} \tag{3.10}$$

$$G_1(v) = (n \cdot v)[((n \cdot l)(1 - \alpha) + \alpha)] \tag{3.11}$$

$$G_1(l) = (n \cdot l)[((n \cdot v)(1 - \alpha) + \alpha)] \tag{3.12}$$

In this section, we only explain the BRDF and several representative shading models here. For more information, please refer to Real-Time Rendering [29] and Physically Based Rendering: From Theory to Implementation [25].

3.2.2 Multi-layer BRDF

In the real world, most materials have more than one layer, such as wooden floors. The wood itself is a rough diffuse surface, but wooden floors are always coated with a layer of specular reflective transparent paint. In addition, plastic, marble, and porcelain also have similar multi-layer surfaces. It is hard to render multi-layer material, previous works in this area tried to solve this problem in different ways. Among them, the method proposed by Weidlich and Wilkie [5] is a simple but still physically based way to simulate smooth and rough multi-layer surfaces. This model simplifies the problem in the following ways.

1. **Thin layer:** The horizontal extent of any microfacet is considered to be much greater than the thickness of the layer
2. **Single-point sampling:** All lights generated by the surfaces of all layers will exit at the original point of incidence.
3. **Single-point refractions:** Refraction rays that are generated for the computation of the entire BRDF, are assumed to meet at a single point on the next layer interface.
4. **Non-scattering medium:** All the scattering is caused by the interactions at the surfaces of layers.

The first simplification assumes that a ray of light entering a microfacet exits through the same microfacet, simplifying both the evaluation and sampling processes. Simplifications two and three, in addition to facilitating evaluation and sampling, ensure that the model remains a single-point reflecting function (BRDF) rather than evolving into an areal sub-surface scattering function (BSSRDF), which would be indispensable for a full light transport simulation. The fourth simplification prohibits volumetric scattering within the medium, avoiding the significant complexity this would introduce.

With these simplifications, we can combine any number of layers of BRDF into a single BRDF to simulate multi-layer BRDF.

Figure 3.4 presents a sketch of the simplified reflection and ray propagation geometry. When examining how light interacts with layers within a surface, we focus on each small, reflective part of the surface, known as a microfacet. These microfacets are significantly larger than the thickness of the layers they rest on. This implies two things:

1. When light strikes one of these microfacets, it will reflect back in the same direction it entered. This simplifies the analysis, as we do not need to account for the light interacting with other nearby microfacets.
2. If light penetrates deeper and is reflected from a layer beneath the surface, it will still come out of the same point from which it first entered, regardless of the angle from which it came out. The path of a ray of light is determined by geometrical principles, similar to representing the trajectory of a ray of light by a straight dashed line.

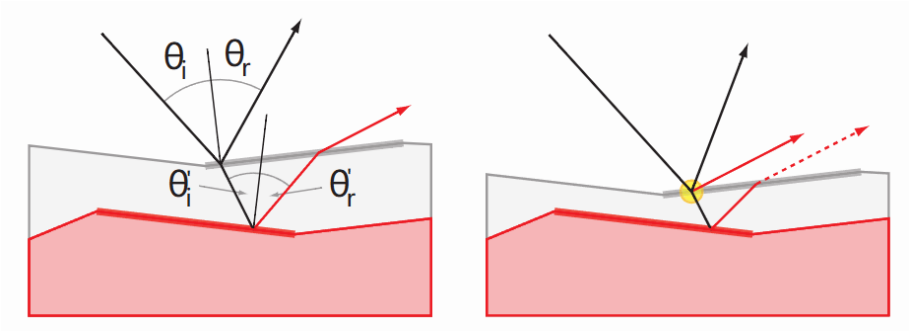


Figure 3.4: Sampling configuration [5]

3.3 Thin-film interference

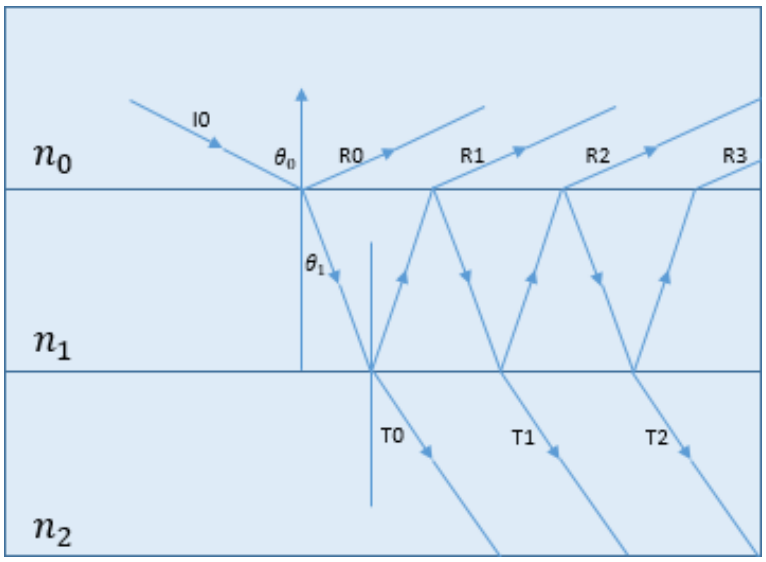


Figure 3.5: Light paths in a single thin-film surface

The iridescence effect in snake skin is produced by specialized cells called iridophores, located in the upper layer of the skin's structure [4]. This effect arises through a phenomenon known as thin-film interference, which occurs when multiple reflections take place within a parallel dielectric film. The thickness of this film is roughly equivalent to the wavelength of the incident light.

Consider a parallel film with thickness d and an index of refraction (IOR) n_1, situated between two media with IORs n_0 for the top layer and n_2 for the bottom layer. When incident light characterized by wavelength λ and angle θ interacts with the film's surface, it undergoes reflection or refraction in accordance with the Fresnel equations. As light transmits through the film, it may undergo multiple reflections before emerging. Upon exiting, the light is dispersed into numerous beams, either from the top or bottom surface of the film.

Each interaction of light with the film's surface results in attenuation due to Fresnel

reflection, occurring at the interface between layers with indices n_i and n_j. Moreover, as light propagates within the film, it incurs a phase delay proportional to the distance traveled.

When light exits the film, whether by reflection from the top or transmission from the bottom, the emergent parallel light beams possess varying phases. These differences are a consequence of the optical paths traversed, leading to constructive or destructive interference. The nature of this interference is contingent upon the incident angle and the wavelength of the light, resulting in the vibrant iridescence observed in snake skin.

This explanation simplifies the complex interactions that contribute to the iridescent appearance, which can include factors such as the snake species, the specific structure of the iridophores, and the environmental lighting conditions.

3.4 Procedural texture generation

Procedural texturing is a technique in computer graphics that generates textures algorithmically, providing a flexible and efficient means to create intricate and detailed surface patterns without the need for pre-made images. This approach employs mathematical functions to produce a wide variety of textures that can be dynamically adjusted through parameters, ensuring consistency and allowing for infinite detail without requiring significant storage. This method is widely utilized in games, visual effects, and architectural visualization to create realistic surfaces. In the context of snake skin and scales, procedural techniques can be used to generate skin pattern textures, scale shapes, and other textures necessary for shading models.

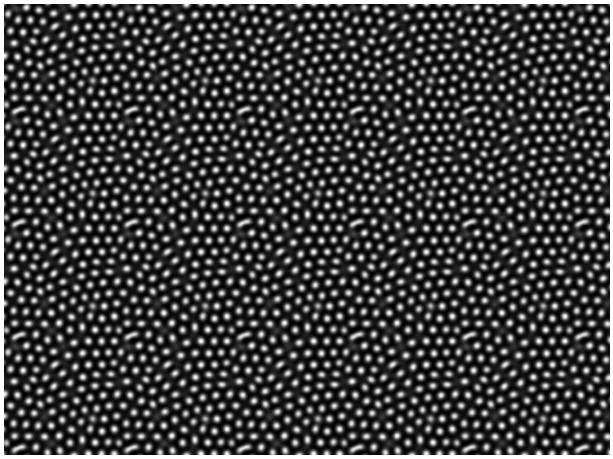


Figure 3.6: Illustration of the reaction-diffusion texture [30]

Reaction-diffusion

Reaction-diffusion system is a way to generate textures. The system simulates complex chemical reactions within fluids, resulting in intricate patterns. Reaction-diffusion draws inspiration from biological processes like morphogenesis (the development of an organisms form). It mimics natural phenomena found in animals,

such as their distinctive markings (think shells, fish, and wild cats). However as Pinheiro and Walter [6] mentioned the reaction-diffusion method proposed by Turk [31]and [32] are too generic, and it is unclear how to apply them to our domain (snakes) such that it covers all, or most of, snake skin patterns.

Noise functions

Perlin noise is a widely used noise created by Ken Perlin. It is generated by blending gradients that are evenly spaced in a grid. By adjusting the spacing of these gradients, we can control the texture's coarseness. Additionally, if the width and height of the image are exact multiples of the cell spacing, the resulting image will tile seamlessly [33].

Worley noise is also a widely used noise that can be used to generate snake skin textures [6]. It was introduced by Steven Worley in 1996, also known as Voronoi noise or cellular noise. Worley noise texture is created by randomly distributing feature points (seeds) in 2D or 3D space, often organized in grid cells. Then, for any location in space, compute the distances Fn to the n-th closest seed (e.g., the second-closest seed). Then use combinations of these distances to control the color information. The noise function W(x) is a vector of distances, which can be combined to produce a color [34].

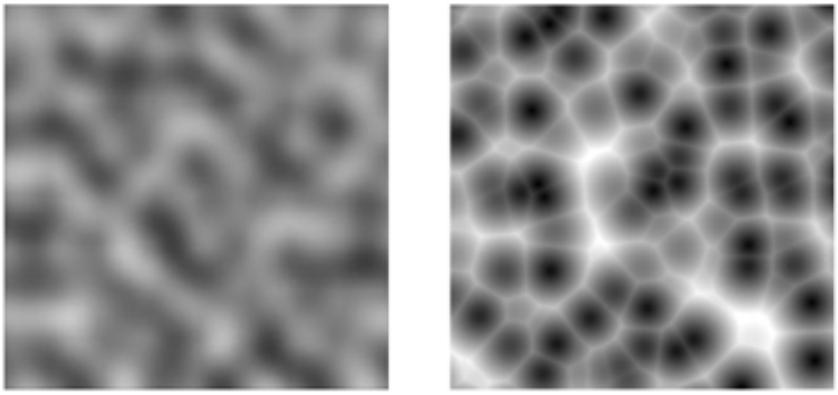


Figure 3.7: Perlin noise and Worley noise [35]

Fractal noise is also an important noise. It is created by adding basic noises(such as Perlin noise) of different frequencies and amplitudes. By adjusting the parameters of the creating process, we can obtain noise with different patterns. In our work, we use fractal noise in many steps of procedural texture generation.

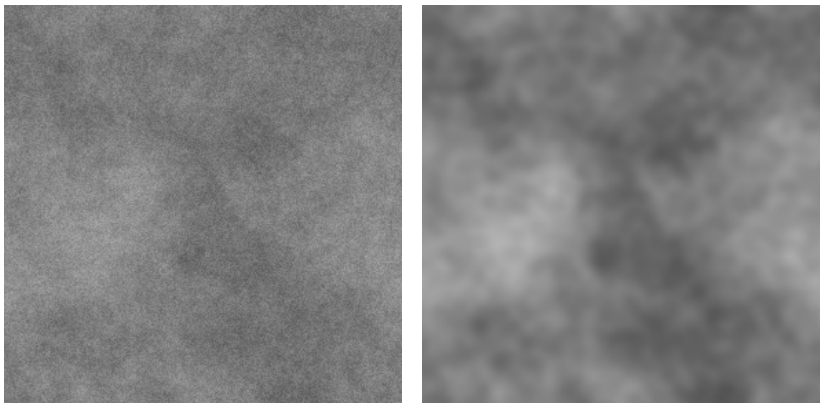


Figure 3.8: Fractal noise with different parameters

4

Methods

This chapter provides a detailed description of the snake skin model we utilized. It covers the procedural generation process for various types of snake skin textures, the acquisition of multi-layer BRDF, and the rendering process for snake skin using the obtained textures and multi-layer BRDF.

4.1 Modeling snake skin structure

As explained in the description of snake skin structure in the Theory chapter, the outermost appearance layer of the snake skin can be modeled as three sub-layers, as illustrated in Figure 3.3:

- The outermost iridescent layer
- The intermediate absorption layer
- The bottom diffuse reflection layer

We employ this snake skin model and provide the parameters for each layer in Table 4.1.

Parameter	**Definition**
n_0	refractive index of the environment
n_1	refractive index of the top(iridescent) layer
n_2	refractive index of the intermediate(absorption) layer
δ_0	thickness of the iridescent layer
δ_1	thickness of the absorption layer
σ_0	absorption coefficient of the absorption layer
k_d	albedo of the bottom(diffuse) layer

Table 4.1: Parameters of the snake skin model

4.2 Procedural generation of snake skin textures

Due to the requirements of the shading method we employ, in addition to the color map, normal map, and roughness map required by the basic PBR process (e.g., Metal-Roughness workflow), we also need an iridescent layer thickness distribution

map to render the iridescent effect on the snake skin surface. Creating these textures by hand is challenging, and the iridescent layer thickness distribution map must be synchronized with the normal map and height map; therefore, we choose to generate all the textures procedurally.

We aimed to ensure our snake skin shading model could be used in a modern game engine or renderer and be easy to configure. Thus, we endeavored to make it compatible with the basic PBR pipeline familiar to artists, and implement all generation logics as a node graph in Substainer Designer, which is a texture creation tool easy to configure by artists. Consequently, we will generate five textures:

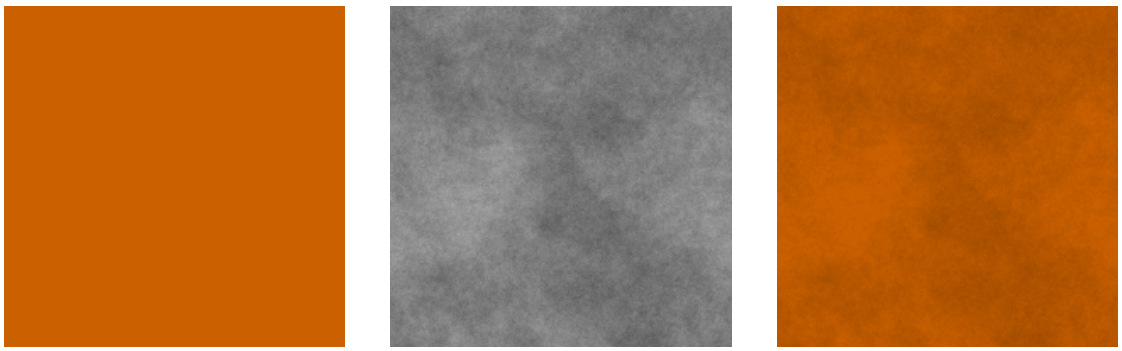
- Color map: Describes the pattern of the snake skin.
- Height map: Describes the surface vertex offset caused by scales.
- Normal map: Describes the normal distribution of the snake skin surface and scales.
- Roughness map: Describes the roughness of the snake skin surface.
- Iridescent layer thickness map: Describes the film thickness distribution of the equivalent thin film provided by the outer iridescent layer.

The following subsections describe the generation process of all textures.

4.2.1 Color map

As described in the patterns of snake skin described in the theory chapter, we procedurally generate them respectively.

4.2.1.1 Base color

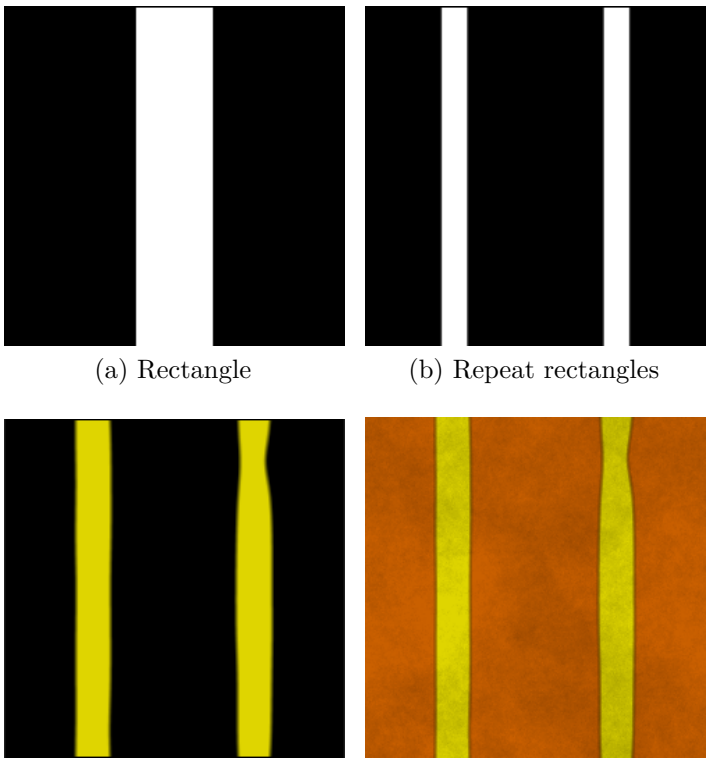


(a) Pure color (b) Fractal noise (c) Color with noise

Figure 4.1: Base color

For the base color, we use one solid color map. Since snake skin will not have a perfect solid color, we add a layer of fractal high-frequency noise to simulate the color variations.

4.2.1.2 Stripe



(a) Rectangle (b) Repeat rectangles

(c) Add noise and color (d) Blend with base color

Figure 4.2: Long stripe pattern

The stripe pattern consists of one or several long stripes on both sides of the body. We first obtain a single stripe by translating and scaling a square. Then the stripe is repeated several times. After that, we blend the stripes with noise and apply a binary threshold to create edge distortion. Finally, we blend the result with the base color to get the final stripe pattern.

4.2.1.3 Horizontal stripe

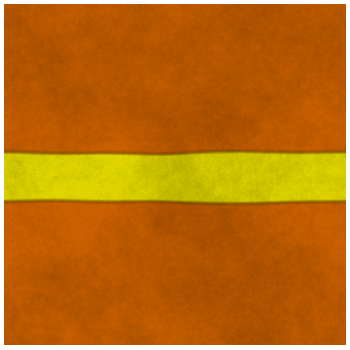
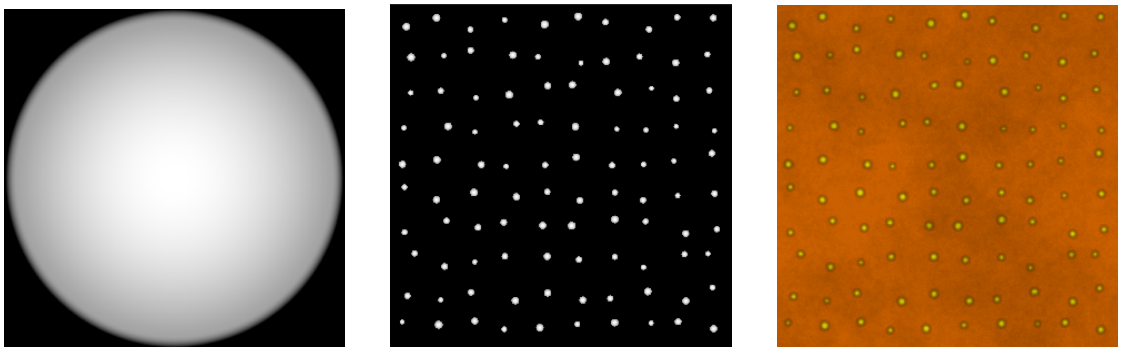


Figure 4.3: Horizontal stripe pattern texture

The horizontal stripe pattern is similar to the stripe pattern. We rotate the stripe pattern by 90 degrees to create the horizontal stripe pattern.

4.2.1.4 Point

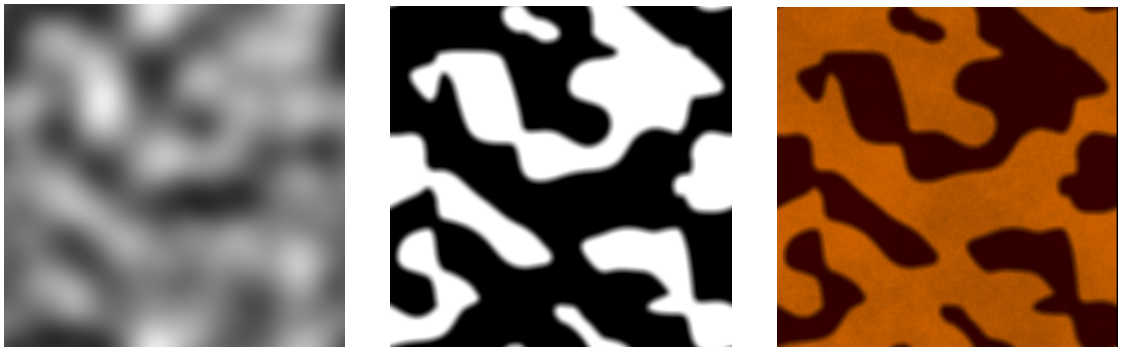


(a) Circle (b) Points (c) Point pattern result

Figure 4.4: Point pattern

The base shape of the point pattern is a circle. We directly tile it to create the points pattern. When tiling, we can adjust the tiling parameters to make the points have slightly different positions and sizes. After tiling, we blend the result with the noise map to create edge distortion.

4.2.1.5 Irregular

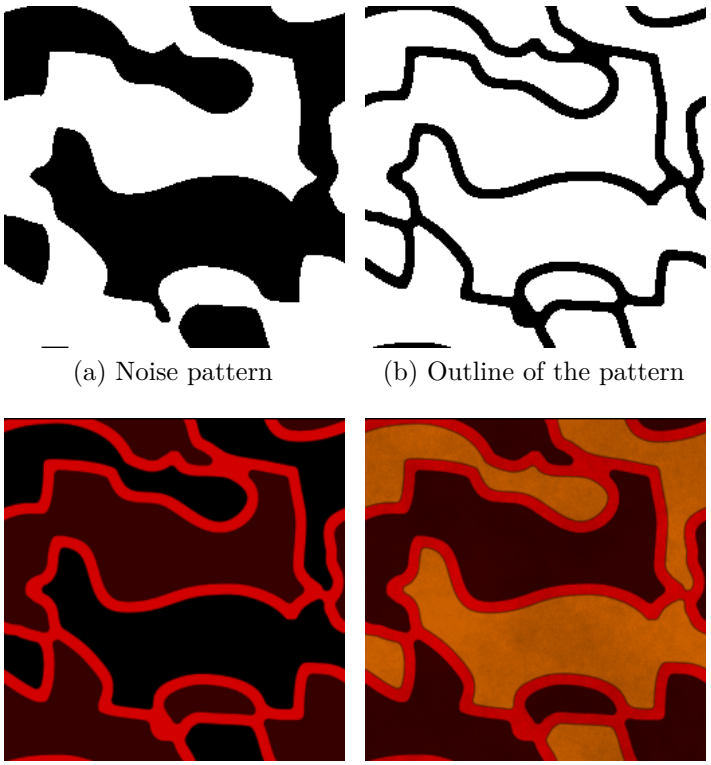


(a) Perlin noise (b) Binary threshold and blurred (c) Irregular pattern

Figure 4.5: Irregular pattern

For some irregular patterns, we generate tileable noises as base shape input. Then, we distort the edges of the generated patterns by blending the result with high-frequency noises. After that, we use a threshold to binarize them. Finally, we use a filter to blur them to create smooth edges. Using different noise maps(e.g., Perlin noise or Worley noise) and parameters can achieve various pattern styles. This generation method includes randomness. For users aiming to reproduce the specific irregular patterns of a certain snake more accurately, they can first draw a rough shape, and then use this procedural process to generate its outline and distortion.

4.2.1.6 Outline



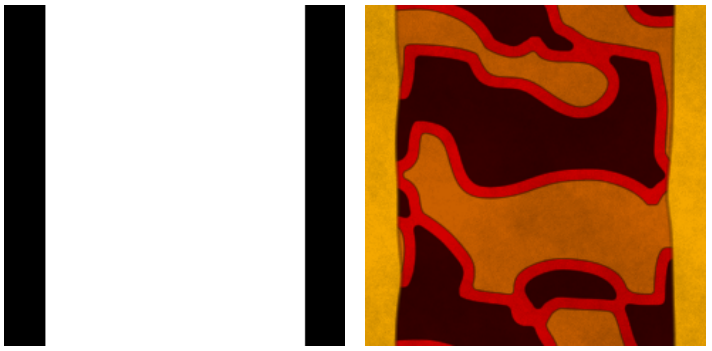
(a) Noise pattern (b) Outline of the pattern

(c) Noise pattern with outline (d) Noise pattern result

Figure 4.6: Outline pattern

Patterns may have an outline with a certain width. We use edge detection to detect the edge of the patterns to get a grayscale edge map. Then we blur and binarize the edge map to adjust the strength of the edges. Finally, we adjust its color and blend the outline with the pattern to obtain an outlined pattern.

4.2.1.7 Belly



(a) Bottom mask (b) Color texture with bottom

Figure 4.7: Color texture

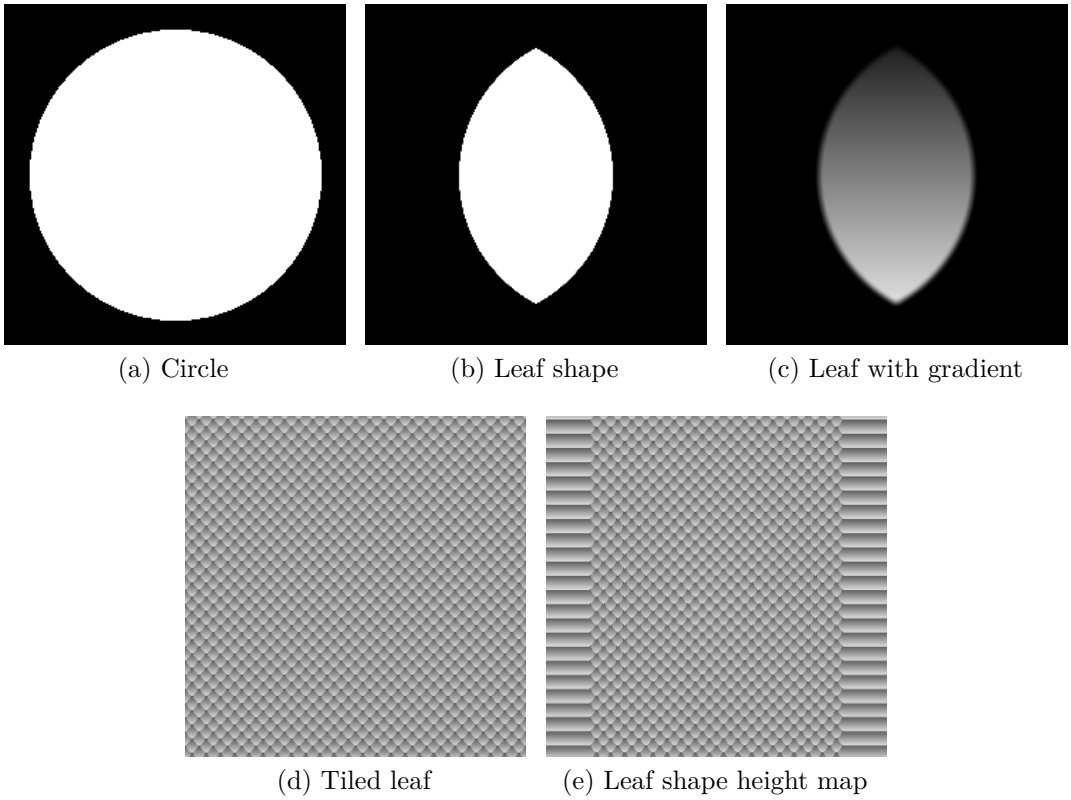
For some snakes, the skin color on the belly is different from the skin on the back, usually lighter or white. This can be achieved by generating a belly color map and blending it with the snake skin color map.

After creating the pattern, we blend it with the base color map to get the final color map of the snake skin.

4.2.2 Height map

Snake skin is covered by scales, we use a height map to store the relative height of the scales. The distribution of height corresponds to the type of scale. To simulate different types of snake skin, we generated three types of scales: leaf, diamond, and hexagon.

1) **Leaf**



(a) Circle (b) Leaf shape (c) Leaf with gradient

(d) Tiled leaf (e) Leaf shape height map

Figure 4.8: Leaf shape height map

We translate and blend two circular shapes to create the leaf shape. Then, we blur the result to smooth the edges. Finally, we blend it with a gradient map to simulate the angle of the scales, thereby obtaining the leaf-shaped scale.

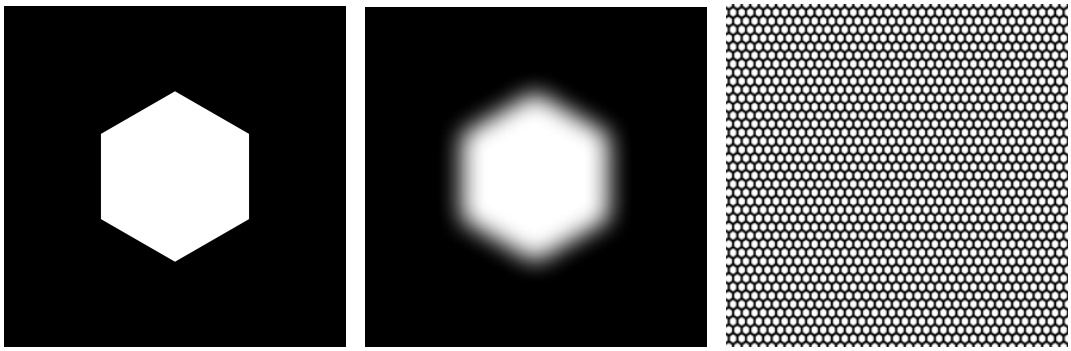
2) **Diamond**

(a) Diamond shape (b) Diamond shape scale (c) Diamond tile map

Figure 4.9: Diamond shape tile map

Similar to the process of creating a leaf scale height map, here we use a rhombus as the base shape.

3) **Hexagon**

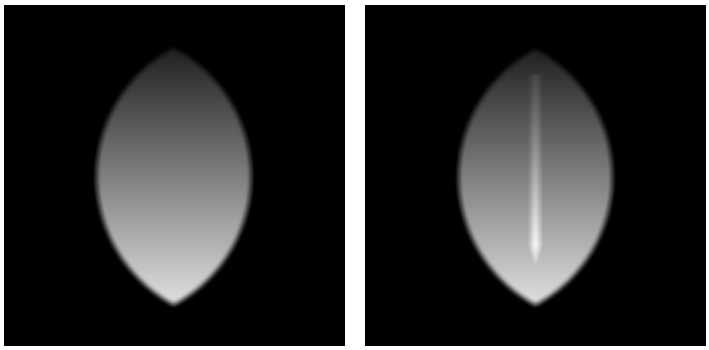


(a) Hexagon (b) Hexagon scale (c) Hexagon height map

Figure 4.10: Hexagon shape tile map

Similar to the leaf shape scale, but we use 1 hexagon as the base shape. The gradient map that hexagonal scales blend with is different from leaf and diamond scales.

4) **Keel**



(a) Leaf shape scale (b) Keeled scale

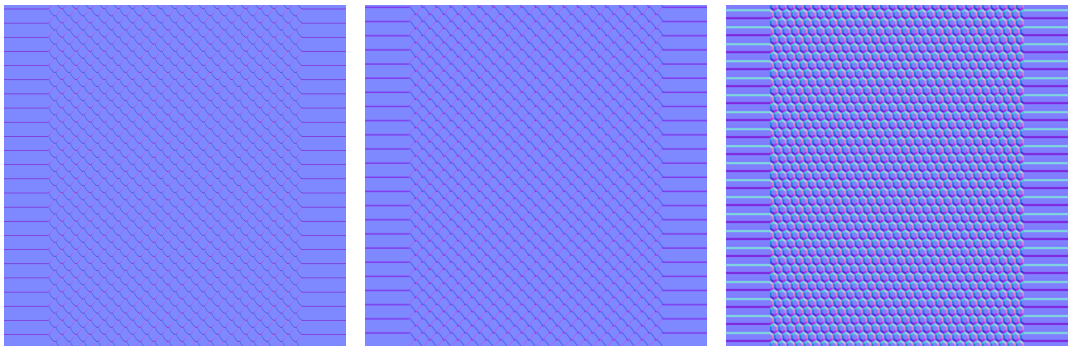
Figure 4.11: Keeled scale

Some snakes have keeled scales, which can be achieved by adding a rectangular grayscale gradient shape on top of the scale pattern.

5) **Tiling**

After obtaining the snake scales, we need to tile them to create the scale texture. For leaf shapes and diamonds, we tile them in an interleaved pattern. For hexagons, we tile them in a non-overlapping manner. The scales on the snake's belly differ from those on its back; they are generally longer and can be seen as normal scales stretched from the middle. This effect can be achieved by stretching the resulting scale map from the center. During tiling, we can adjust the size, position, and orientation of the pattern randomly to produce more natural results.

4.2.3 Normal map



(a) Leaf normal map (b) Diamond normal map (c) Hexagon normal map

Figure 4.12: Normal maps

We need surface normal that correspond to the height map, so we must generate the surface normal using the same method as the height map. In this process, we directly use the obtained height map by converting it into a surface normal map in

tangent space based on the height map. This is done by using a height to normal node in Substainer Designer.

4.2.4 Roughness map

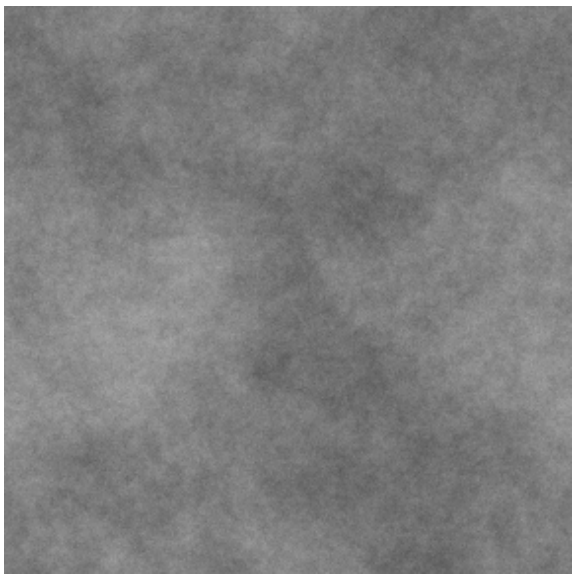
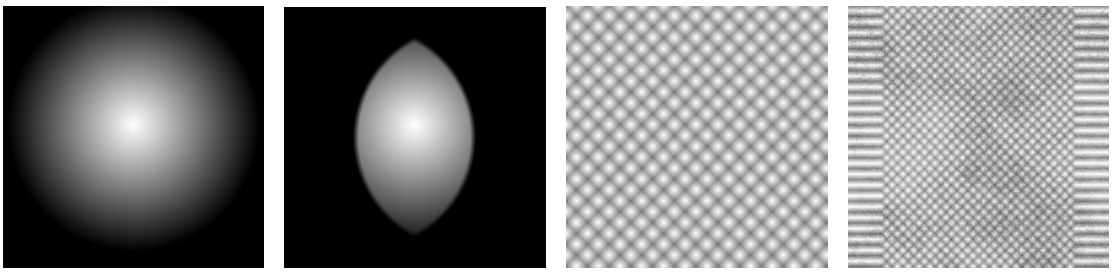


Figure 4.13: Roughness texture

The roughness or glossiness of the surface can be described by a roughness map. We use a high-frequency noise map to simulate the fine details of the surface. By adjusting its parameters (e.g., frequency or texture alpha), we can generate snake skin with either smooth or rough scales.

4.2.5 Iridiscent layer thickness distribution map



(a) Radial (b) Iridiscent thickness of a scale (c) Tiled thickness map (d) Iridiscent thickness map

Figure 4.14: Iridiscent thickness map

Since the thickness of the iridescent layer at a certain point on the snake skin is related to its position on the scales, we can adjust the value range and add noise

based on the height map that describes the shape of the scales to simulate thickness variations in the iridescent layer.

We blend a radial gradient map with the scale to simulate the variation in iridescent layer thickness on a single scale. This blended map is then tiled like the height map to create the complete map. Finally, we add high-frequency fractal noise to enhance its natural appearance.

4.3 Multi-layer BRDF Derivation

As described in our modeling of snake skin, the reflection of the snake skin surface is mainly composed of the specular reflection of the iridescent layer and the diffuse reflection of the base layer.

We start by calculating the reflection term of the iridescent layer, which is the main source of the iridescent effect on the snake skin surface. Then, based on the structure of the snake skin, we derive the formula for the multilayer BRDF. Finally, we combine them to obtain the complete multi-layer BRDF for snake skin rendering.

4.3.1 Iridescent reflection

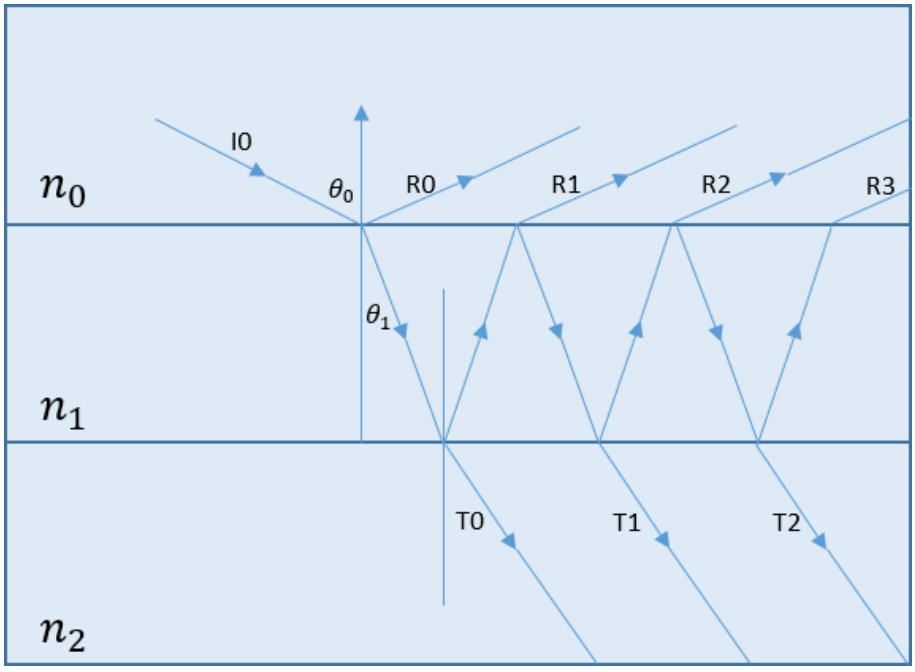


Figure 4.15: Light paths in iridescent layer

First, we determine the energy proportion of light reflected by the film layer based on the Fresnel term of the film layer. Figure 4.15 shows the path of light reflected and refracted at the interface of the iridescent layer.

According to the Fresnel equation, we have:

$$r_{s,i|j} = \frac{n_i cos\theta_i - n_j cos\theta_j}{n_i cos\theta_i + n_j cos\theta_j} \tag{4.1}$$

$$r_{p,i|j} = \frac{n_j cos\theta_i - n_i cos\theta_j}{n_i cos\theta_j + n_j cos\theta_i} \tag{4.2}$$

$$t_{s,i|j} = \frac{2n_i cos\theta_i}{n_i cos\theta_i + n_j cos\theta_j} \tag{4.3}$$

$$t_{p,i|j} = \frac{2n_i cos\theta_i}{n_i cos\theta_j + n_j cos\theta_i} \tag{4.4}$$

Assume that the amount of s-polarisation is equal to the amount of p-polarisation. For the transmitted light $T0$, we can see that its transmittance is $t_{01}t_{12}$. For $T1$, it transmits once from layer 0 to layer 1, reflects from the layer 1 to layer 2 interface, then reflects again from the layer 1 to layer 0 interface, and finally transmits via the layer 1 to layer 2 interface. Its transmittance is $t_{01}t_{12}r_{01}r_{12}$. From this we can conclude that for transmitted light T_k its transmittance is:

$$T_k = t_{01}t_{12}(r_{01}r_{12})^k \tag{4.5}$$

For all transmitted light $T0, T1...T_k$ we need to find the sum of their transmittances. Since the phase of light waves changes as they propagate through the medium, these waves can have constructive or destructive effects on each other depending on their phase. Taking into account the phase difference caused by light propagation and light at the refractive interface, the phase of T_k is:

$$\phi_k = k(\frac{2\pi}{\lambda}(2n_1 d cos\theta_1) + \Delta) \tag{4.6}$$

In Formula (4.6), λ is the wavelength of the light, d is the thickness of the iridescent layer, Δ represents the phase changes when light is reflected off a surface:

$$\Delta_{ij} = \begin{cases} 0 \text{ if } n_i > n_j \\ \pi \text{ if } n_i < n_j \end{cases} \tag{4.7}$$

Then add the transmittance of all the transmitted light to give the total transmittance. According to physical optics, the sum of the intensities of two coherent lights can be calculated by taking the square of their complex amplitudes. This gives the total transmittance:

$$T = | \sum_{k=0}^{\infty} T_k e^{i\phi_k} |^2 \tag{4.8}$$

This is a geometric series, we can then simplify it to:

$$T = \frac{t_{01}^2 t_{12}^2}{r_{01}^2 r_{12}^2 - 2r_{01}r_{12}cos\phi + 1} \tag{4.9}$$

As the area of the light and the medium IOR change during the refraction process from medium 0 to medium 1 and from medium 1 to medium 2, it must be multiplied by:

$$\frac{n_2 cos\theta_2}{n_0 cos\theta_0} \tag{4.10}$$

The final total transmittance of the film layer is:

$$T = \frac{n_2 cos\theta_2}{n_0 cos\theta_0} \cdot \frac{t_{01}^2 t_{12}^2}{r_{01}^2 r_{12}^2 - 2r_{01}r_{12}cos\phi + 1} \tag{4.11}$$

Based on energy conservation, we have:

$$T + R = 1 \tag{4.12}$$

We can get the total reflectance of the film layer:

$$R = 1 - T \tag{4.13}$$

We use R term to replace the Fresnel term of the specular part in BRDF to get the specular part in our BRDF:

$$f_{spec} = \frac{DRG}{4(\omega_i \cdot n)(\omega_o \cdot n)} \tag{4.14}$$

4.3.2 Multi-layer BRDF Derivation

We first calculate the direction of light propagation within the film layer using the law of refraction:

$$sin\theta_1 = \frac{sin\theta_0 n_1}{n_0} \tag{4.15}$$

According to the Bouguer-Lambert-Beer law, when light of intensity I travels a distance l in the material, the energy remaining after being absorbed by the medium is:

$$I = I_0 e^{-\sigma l} \tag{4.16}$$

In (4.16), σ is the absorption factor of the film layer, d is the thickness of the film layer.

Assuming that the angle at which the light interacts with the second layer (absorption layer) is θ_1, and the thickness of the film is d, then the propagation distance of the light from the film surface to the surface of the second layer is $\frac{d}{cos\theta_2}$.

The energy ratio of the light reaching the surface of the base layer is:

$$T_{in} \cdot e^{-\sigma \cdot \frac{d}{cos\theta_2}} \tag{4.17}$$

When light reaches the surface of the base layer, it will interact with the surface and be reflected. We need to multiply by the BRDF of the surface. Therefore, the energy ratio of the light leaving the surface of the base layer is:

$$T_{in} \cdot e^{-\sigma \cdot \frac{d}{cos\theta_2}} \cdot f_{base}(P_1, \omega_i, \omega_o) \tag{4.18}$$

When light leaves the surface of the base layer, the propagation distance from the surface of the base layer to the bottom interface of the iridescent layer is $\frac{d}{cos\theta_3}$. Then the energy ratio of light leaving the surface of the base layer is:

$$T_{in} \cdot e^{-\sigma \cdot d \cdot (\frac{1}{cos\theta_2} + \frac{1}{cos\theta_3})} \cdot f_{base}(P_1, \omega_i, \omega_o) \tag{4.19}$$

When the light reaches the iridescent layer again, the proportion of light that is transmitted to the outside is determined according to the Fresnel term of the film layer, just like the incident light.

For the BRDF of the base diffuse layer, we choose to use the Lambertian BRDF. Combining the energy directly reflected by the thin film layer and the energy reflected by the base layer, we can summarise the BRDF of the diffuse part as follows:

$$f_{diff} = T_{out} \cdot T_{in} \cdot e^{-\sigma \cdot d \cdot (\frac{1}{cos\theta_2} + \frac{1}{cos\theta_3})} \cdot \frac{k_d}{\pi} \tag{4.20}$$

The final BRDF we obtained is as follows:

$$f_{final} = T_{out} \cdot T_{in} \cdot e^{-\sigma \cdot d \cdot (\frac{1}{cos\theta_2} + \frac{1}{cos\theta_3})} \cdot \frac{k_d}{\pi} + \frac{DRG}{4(\omega_i \cdot n)(\omega_o \cdot n)} \tag{4.21}$$

4.4 Implementation

We have rendered the snake skin in real-time using rasterization and offline using a path-tracing renderer. To ensure a fair comparison, we applied the same textures and shading models in both renderers. In this section, we will briefly introduce our implementation of the snake skin in both rendering methods.

4.4.1 Rasterization

We want our shading model to perform physically based rendering of snake skin in real-time. To better verify the rendering result and the performance of our model, and also to make the model easy to use in practical applications, all our work related to real-time rendering is done on the rasterization rendering pipeline in the Unity engine.

In Section 4.3 we completed the derivation of the reflection and transmission formulas of the iridescent layer. It can describe the effect of the iridescent layer or thin film layer on the highlight reflection. We can apply it to the BRDF by replacing the Fresnel term in the specular part to get a BRDF that supports the rendering of thin film interference effects.

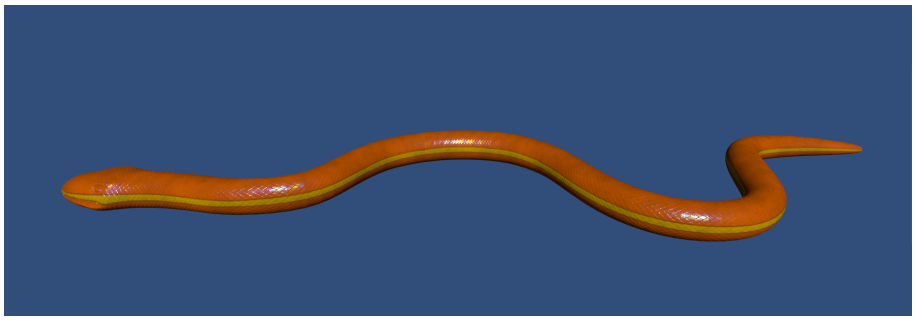


Figure 4.16: Replace the Fresnel term with thin-film reflection term R to render snake skin

After confirming that the iridescent effect caused by thin film interference can be rendered as we expected, we replace the BRDF with a multi-layer BRDF that models the snake skin surface structure and applies the procedurally generated texture in Section 4.2. The render result is shown below.

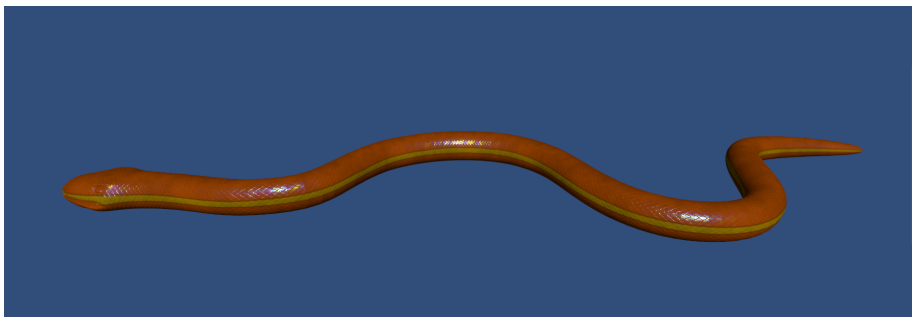


Figure 4.17: Render result of our shading model

4.4.2 Path tracing

To compare the results of real-time rendering, we needed a ground truth appearance. For this reference, we adopted the same shading model with the real-time rendering method and implemented it using our own path tracing renderer.

The rendering loop of the path-tracer consists of three main parts: direct illumination, emission, and indirect illumination. In the direct illumination process, we shoot a shadow ray to each light source in the scene. If no geometry blocks the light, we multiply the radiance from the light with the BRDF and the cosine term. During the emission process, we add the emitted radiance from the intersection. In the indirect illumination process, we add the incoming indirect light by shooting a single ray in a random direction.

To implement the layered material, we added two important materials to the path-tracer: an absorbing material and a thin film material. The absorbing material mixes an absorbing layer with a regular diffuse material using the classical Beer-Lambert exponential transmittance model. The thin film material then mixes the absorbing material with the Microfacet BRDF material using the iridescent reflection factor, following the equation $I \cdot f_r + (1 - I) \cdot f_{absorb}$

5

Analysis and results

In the Methods chapter, we provide a detailed description of our procedural texture generation method and the process of implementing the snake skin shading model. In this chapter, we will present the results of texture generation and snake skin rendering and analyze the impact of different parameters on the results. Furthermore, we test and record the performance of our model.

5.1 Snake skin texture generation

The generation of textures required by the snake skin shading model is achieved through the use of procedural generation. Table 5.2 shows some of the important parameters used to control the generator. Figure 5.1 shows the parameters that users can adjust in the Unity editor. By controlling these parameters, we can create snake skin textures with different appearances.

Parameter	**Definition**
Seed	Random seed of the generator
ScaleType	Leaf shape, diamond or hexagon
ScaleAmount	The amount of scales
ScaleSize	The size of scales
ScaleKeeled	Whether the scales are keeled
PatternType	Stripes, points or irregular
PatternAmount	The amount of patterns
PatternSize	The size of patterns
PatternOutlineSize	The size of pattern outline
EdgeDistortion	Control the distortion strength of pattern edges
BellySize	The width of belly

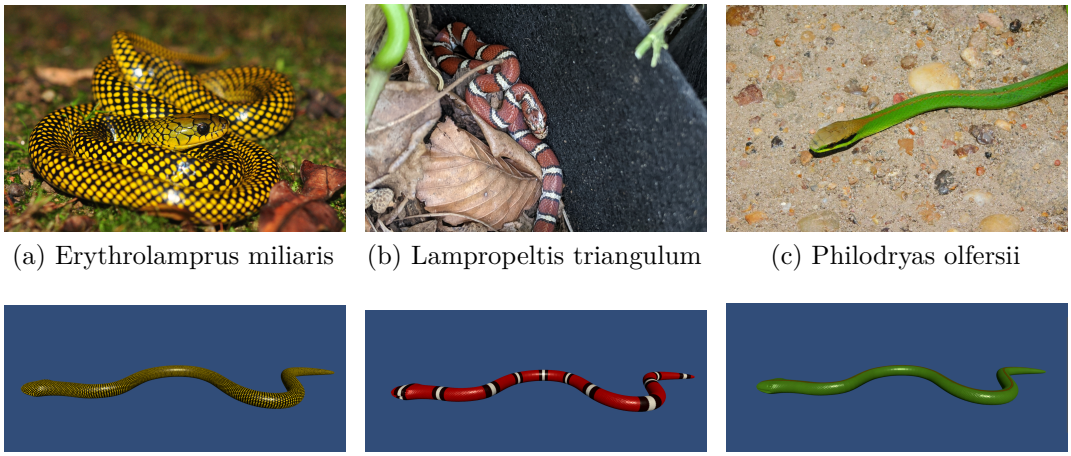
Table 5.1: Parameters of the texture generator

Figure 5.1: Parameters exposed in Unity editor

For texture realism, we adjust the parameters regarding real snakes and generate some snake skin textures. As shown in Figure 5.2, the generated snake skin texture pattern can match the real snake skin texture pattern.

For texture fineness, by adjusting the generation resolution parameter, we can generate maps with resolutions from 1K to 4K, which can meet the requirements of most applications.

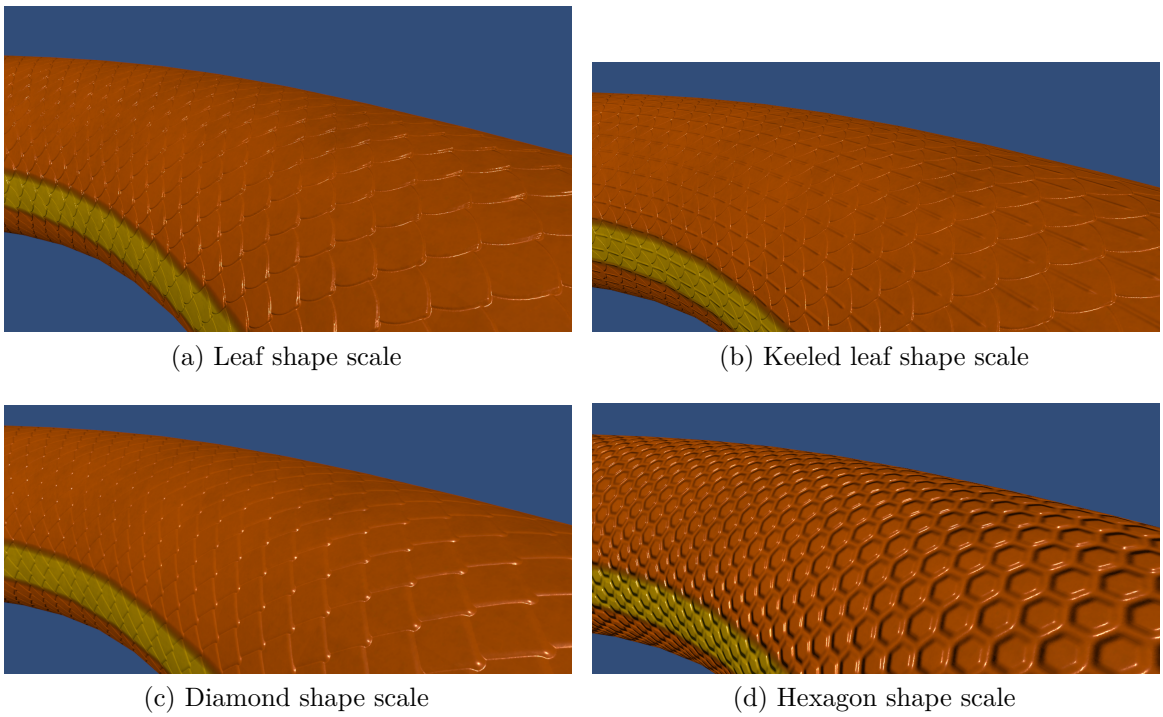
Scales affect the shading of the model through height maps and normal maps. We



(a) Erythrolamprus miliaris (b) Lampropeltis triangulum (c) Philodryas olfersii

Figure 5.2: Reference (top) and textures generated by our generator (bottom) From left to right: Erythrolamprus miliaris [36], Lampropeltis triangulum [37], Philodryas olfersii [38]

generate several different scales regarding real snake scales. The scale shape, size, arrangement, and density can be controlled through parameters.

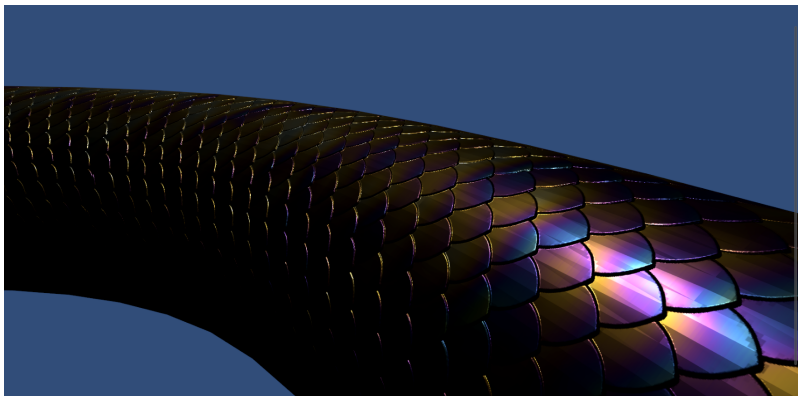


(a) Leaf shape scale (b) Keeled leaf shape scale

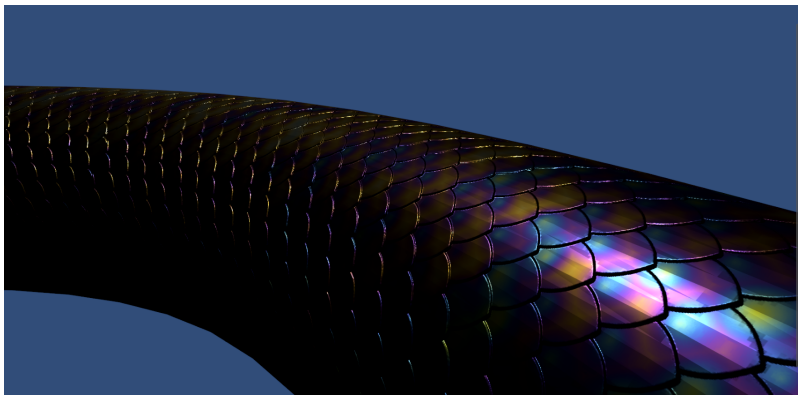
(c) Diamond shape scale (d) Hexagon shape scale

Figure 5.3: Different types of scales

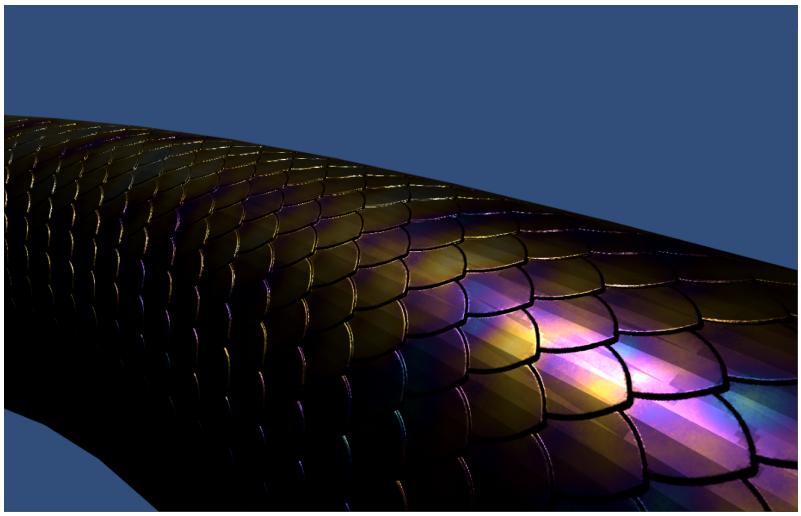
For the iridescent layer thickness distribution map, by adjusting the gradient and noise applied to the scale shape map, we can obtain different distributions of iridescent effects on the scale surface, as shown in Figure 5.4



(a) Vertical gradient



(b) Radial gradient



(c) Radial gradient with fractal noise

Figure 5.4: Different distributions of iridescent effects

In terms of generation speed, our texture generation process is efficient. For a 2K resolution snake skin texture that can be used for our shading model, it takes less than 1 second to generate and export the file in PNG format. This means we can quickly debug and iterate to get the desired results when creating procedural textures.

5.2 Snake skin shading

We use a multi-layer BRDF to simulate the complex light interactions on the snake skin surface. From the multi-layer BRDF formula derived in the Methods chapter, we extract some important variables as adjustable parameters and list them in Table 5.2. We also list some default values of the parameters we used to render the results.

Parameter	Definition	Default
n_0	refractive index of the environment	1.00
n_1	refractive index of the top(iridescent) layer	1.55
n_2	refractive index of the intermediate(absorption) layer	2.00
δ_0	thickness of the iridescent layer	400
δ_1	thickness of the absorption layer	1
σ_0	absorption coefficient of the absorption layer	0.6
k_d	albedo of the bottom(diffuse) layer	-

Table 5.2: Parameters of the snake skin texture generator

5.2.1 Real-time rendering

The base thickness of the iridescent layer will affect its appearance, as changing the thickness will affect the time that light travels in the film, thereby changing the optical path difference(OPD) between the reflected lights, and ultimately changing the iridescent color caused by interference. Figure 5.5, 5.6 and 5.7 shows the results with different iridescent reflection colors after adjusting the base thickness of the iridescent layer.

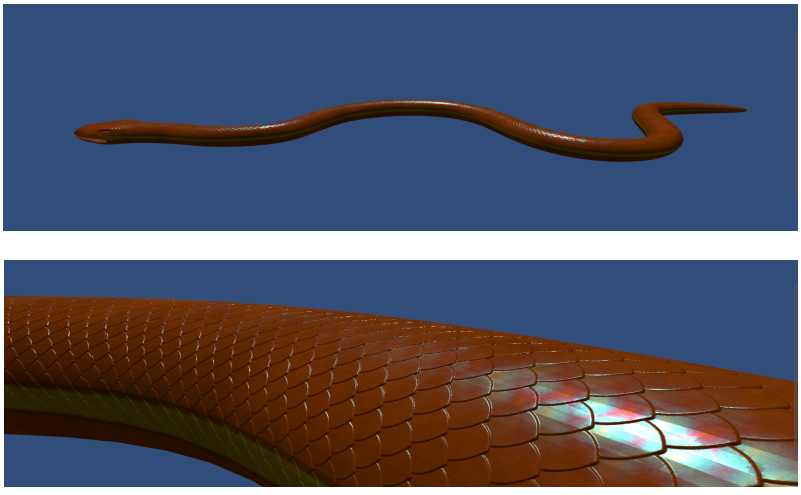


Figure 5.5: Render result of $\delta_0 = 200nm$

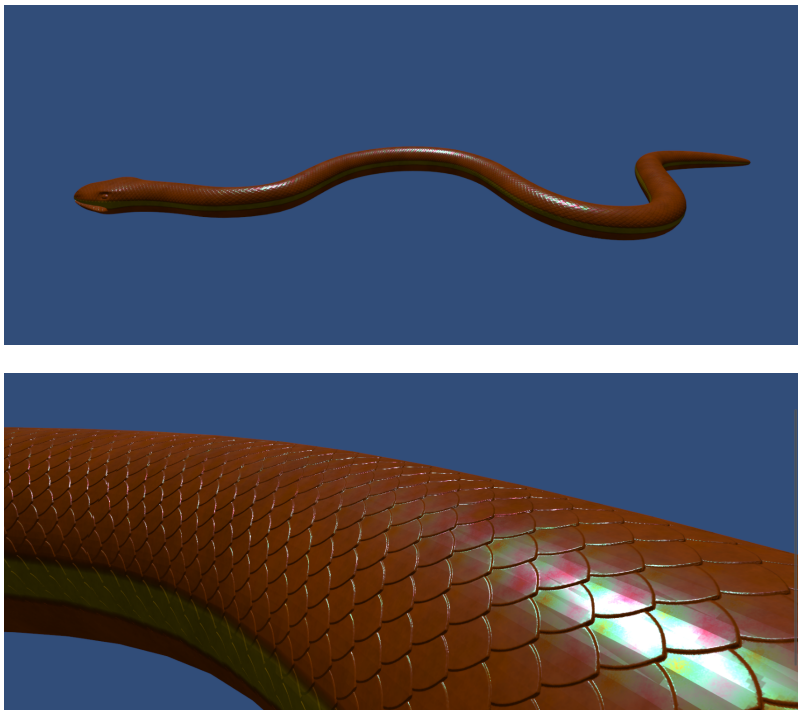


Figure 5.6: Render result of $\delta_0 = 400nm$

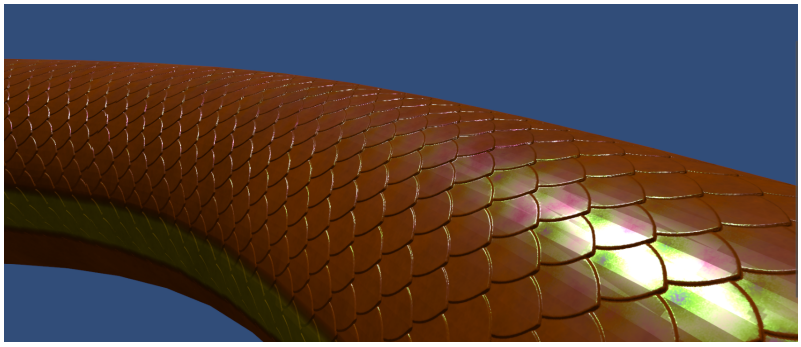


Figure 5.7: Render result of $\delta_0 = 600nm$

If the thickness of the iridescent layer is outside the wavelength of visible light, that is, much smaller or much larger than the wavelength of visible light, the iridescent effect will not appear. In particular, if the thickness of the iris layer is 0, the iridescent reflection will degenerate into ordinary Fresnel reflection. Figure 5.8 shows the corresponding rendering results.

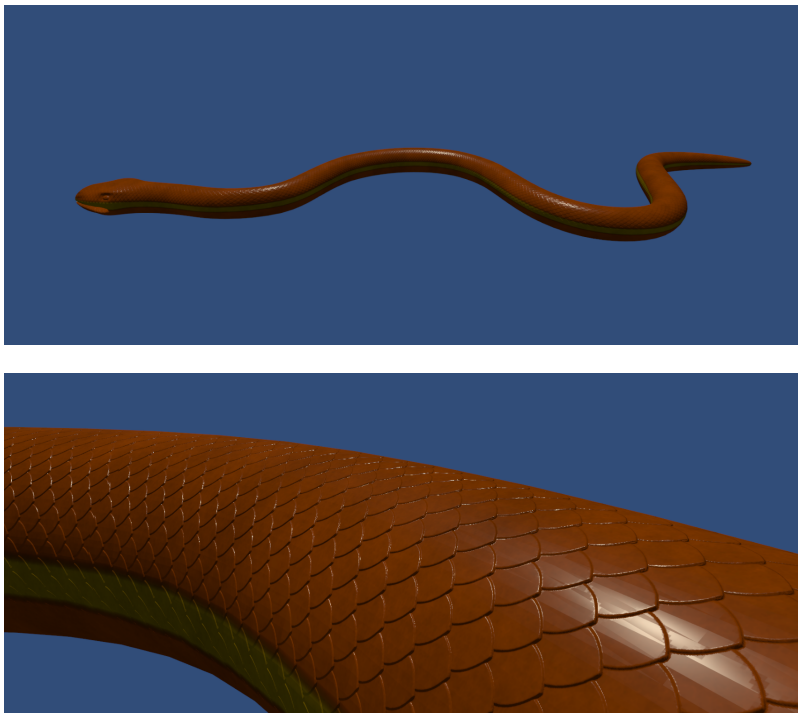


Figure 5.8: Render result of $\delta_0 = 0nm$

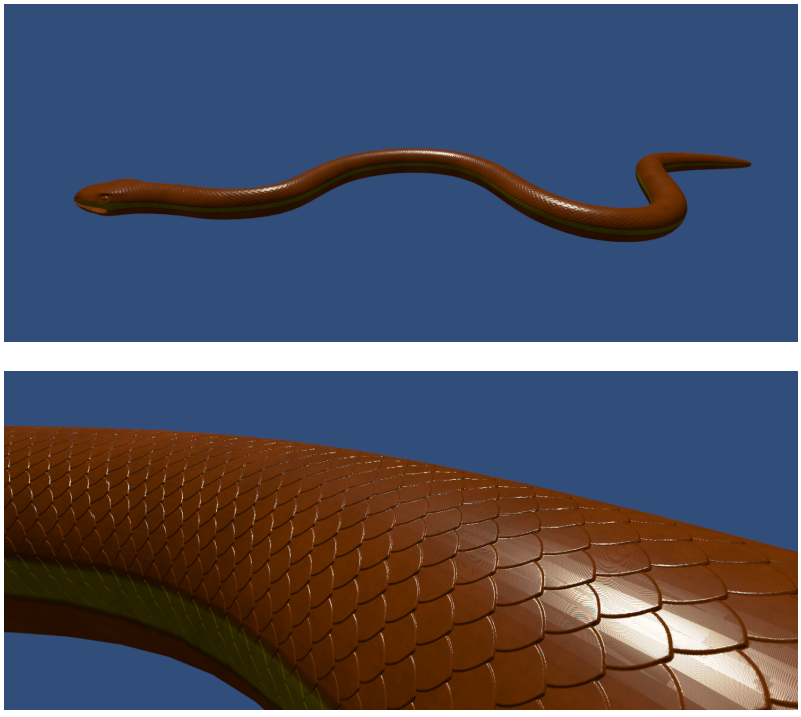


Figure 5.9: Render result of $\delta_0 = 2.4mm$

In Figure 5.8 and 5.9, when the equivalent film thickness of the iridescent layer is zero, there is no iridescence effect. When it is 2.4mm, because the thickness is much larger than the wavelength of the light, there is almost no iridescence effect.

However, due to the precision of the simulation, we can see moiré patterns caused by aliasing.

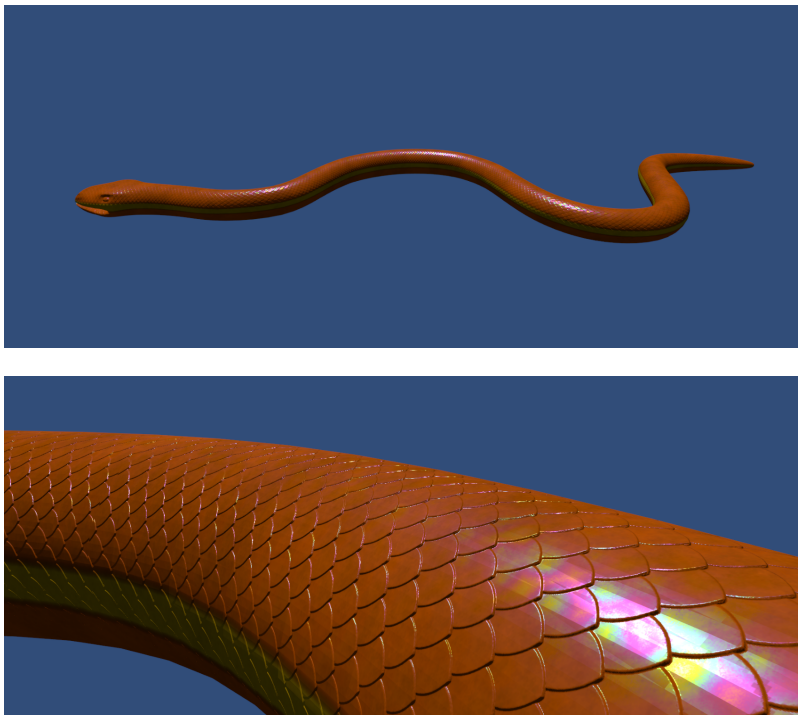


Figure 5.10: Render result of $\sigma_0 = 0.1$

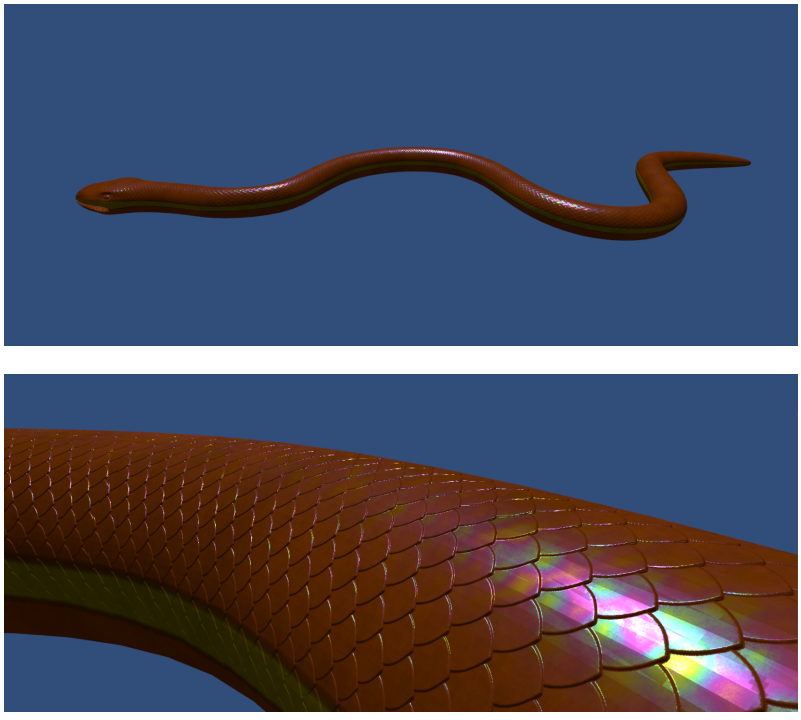


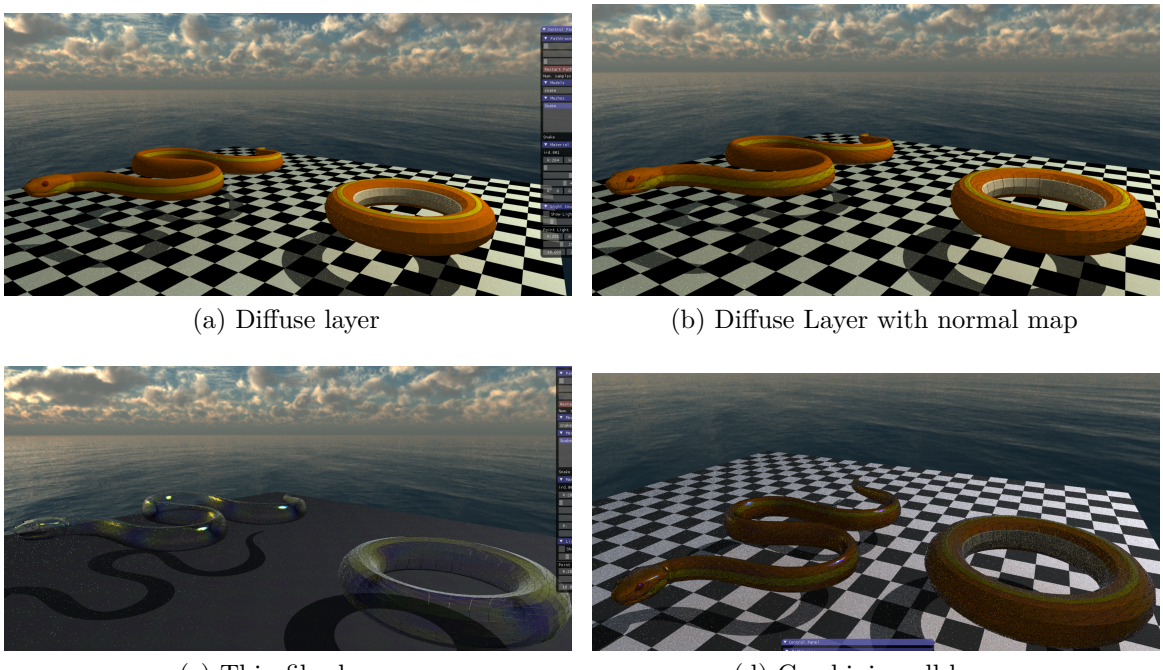
Figure 5.11: Render result of $\sigma_0 = 0.7$

The thickness and absorption coefficient of the absorption layer determines the light absorption effect of the snake skin. For more convenient debugging and configuration, we can combine these two parameters into a single absorption coefficient. As shown in Figures 5.10 and 5.11, increasing the absorption coefficient makes the snake skin darker. We can also extend the absorption coefficient to an RGB vector, allowing the absorption layer to have different absorption capabilities for light of different colors.

5.2.2 Offline Rendering

As mentioned in Section 4.4.2, we have rendered the snake not only in real-time but also using a custom path-tracer, which provides us with a reference by simulating the full light transport for snake skin rendering.

To better demonstrate the three basic layers of the snake skin, we rendered the snake in four stages: the diffuse layer, the diffuse layer with a normal map, the isolated thin-film layer, and all layers combined. The results of these four stages are shown in Figure 5.12.



(a) Diffuse layer

(b) Diffuse Layer with normal map

(c) Thin film layer

(d) Combining all layers

Figure 5.12: Snake skin path tracing results. (a) Geometric models with diffuse material. (b) Adding normal mapping on top of the underlying surface to model the macroscopic appearance of the scales. (c) Replacing the diffuse material with thin-film interference ($IOR = 1.56$, $\delta_0 = 900nm$). (d) Adding an absorption layer (Absorption Coefficient = 0.4) to darker the final appearance.

5.2.3 Validation

5.2.3.1 Performance

In terms of performance, we want our snake skin shading model to support real-time rendering. We tested it in Unity's rasterization pipeline. We use a PC with an Intel Core i5-10400F CPU and an RTX3070 GPU as a test machine, and perform real-time rendering tests in Unity's editor environment. We use our shader to render different numbers of snake models with a single directional light source in an empty scene. The results are recorded below.

Model Amount	Render time cost(ms)
0	0.09
1	0.17
5	0.18
20	0.21

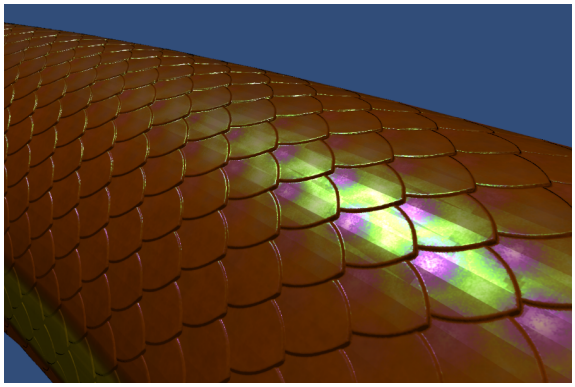
Table 5.3: Performance of the snake skin shading model

The results of the performance test show that under Unity's rasterization pipeline, our shading model can efficiently perform physically-based rendering of snake skin. The FPS can reach the real-time standard(60FPS) and can be used for real-time rendering for various applications.

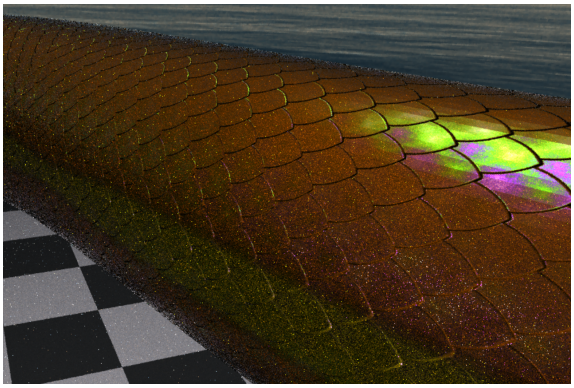
5.2.3.2 Comparison with Offline Rendering

Real-time rendering prioritizes speed and interactivity, often at the expense of some visual fidelity. To validate our real-time rendering algorithm, we compared the output quality of our real-time algorithm against the high-quality, photorealistic images produced by offline rendering. Metrics such as thin film color accuracy with different thicknesses of the iridescent layer, and the light absorption due to different absorption efficiencies of absorption layers were evaluated.

Since the rendering platforms we use for real-time rendering and offline rendering methods have some hard-to-change interference between them, such as the light source, the exact position of the camera and the object, the color space, etc. we only selected these two important factors as elements for comparison see Figure 5.13 and Figure A.1.

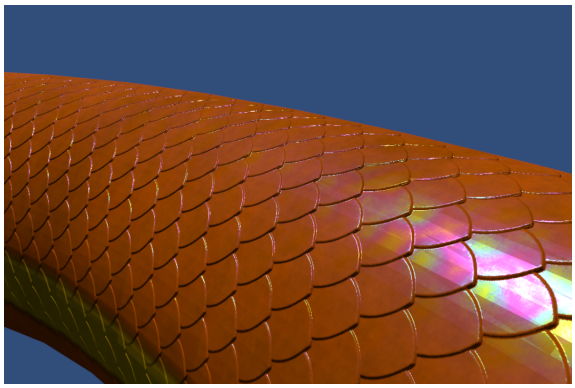


(a) Real-time rendering result $\delta_0 = 600nm$

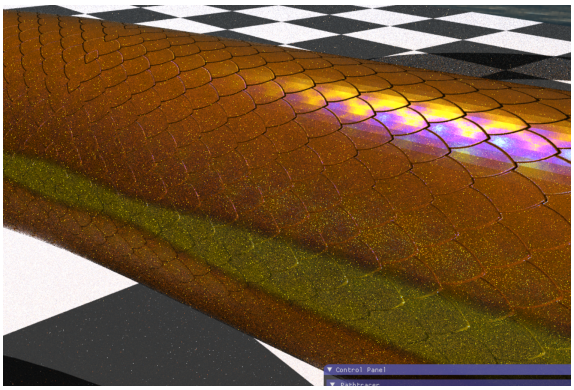


(b) Offline rendering result $\delta_0 = 600nm$

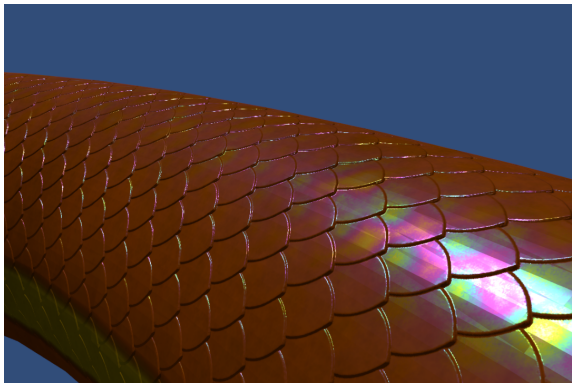
Figure 5.13: Comparison between real-time rendering and offline rendering with same thickness of the iridescent layer



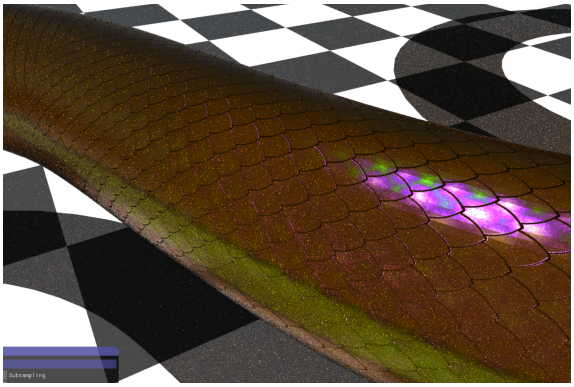
(a) Real-time rendering result $\sigma_0 = 0.1$



(b) Offline rendering result $\sigma_0 = 0.1$



(c) Real-time rendering result $\sigma_0 = 0.7$



(d) Offline rendering result $\sigma_0 = 0.7$

Figure 5.14: Comparison between real-time rendering and offline rendering with different absorption coefficients.

We also compared the rendering results of real-time rendering and offline rendering with thicknesses other than 600nm, as listed in Appendix A.

5.3 Ethical Consideration

As we advance the capabilities of real-time, physically based rendering techniques for applications like snake scale rendering, it is crucial to reflect on the ethical implications of such work. After careful consideration, we believe that our real-time, physically based rendering technique for snake scales does not inherently present any immediate social or ethical issues. Nevertheless, as realistic rendering technologies continue to advance, there is the potential for these techniques to be used in creating deceptive or misleading content. Although the chances are slim that the highly realistic rendering of snake scales will be directly misused, we remain optimistic that it will be employed responsibly in applications such as games or educational materials.

Moreover, the increasing realism of virtual environments, including the detailed rendering of natural elements like snake skins, might encourage a preference for virtual experiences over real-world engagements. This trend could negatively impact both physical and mental health by reducing interaction with the natural environment. Although these concerns are relevant to the broader field of computer graphics rather than the specific focus of this study, they warrant attention.

It is crucial to weigh these concerns against the substantial benefits that realistic computer graphics offer across various domains, such as education, conservation, and entertainment. Accurate visual representations can foster a better understanding and appreciation of the natural world, support conservation efforts, and enhance educational outcomes. Therefore, while remaining vigilant about potential misuse, we should also recognize the positive contributions that advanced rendering technologies can make.

6

Conclusion

In our work, we model the complex multi-layer structure of the snake skin surface as a multi-layer BRDF and integrate it into a physically based rendering method. The top layer is a thin film layer, which simulates the iridescence effect caused by thin film interference. The middle layer is an absorption layer, which simulates the absorption of light by pigment cells in snake skin. The bottom layer is a diffuse reflection layer, which simulates the diffuse reflection of light by the snake skin base. The results of using this lighting model to render snakes under different parameters and configurations are presented, demonstrating that our model can match the appearance of real-life snake skin to a certain extent.

Furthermore, a procedural texture generation model has been implemented that is capable of synthesizing a variety of textured skin patterns of snakes and efficiently providing the specific texture required for the proposed model. The generator is derived from representations of the appearance of snakes, using image processing to generate textures. The results show good visual similarity to the real snake skins.

6.1 Limitaion and Future Work

In our work, we model the iridescent layer as a thin film and use thin film interference to simulate the iridescent effect. However, in addition to interference, light diffraction also can cause visual effects similar to the iridescent. According to the micro structure of the snake skin, since the size of the small flakes inside the iridescent layer and the size of the gaps between the flakes are similar to the order of magnitude of the wavelength of light, the light will be diffracted when passing through the gaps between the flakes, which can also cause iridescence effect. In the future, we can model this structure to integrate the iridescence caused by diffraction effects into the shading model.

In order to simplify the problem of modeling multi-layer BRDF, we employed the simplification method proposed by Weidlich and Wilkie [5]. At present, there are several other methods available for modeling multi-layer BRDF [17][39]. Although these methods can achieve good rendering effects, they still have performance issues for real-time rendering and difficulties in adjusting parameters. In the future, we can further expand the snake skin model and use a more accurate multi-layer BRDF modeling method to achieve better rendering quality.

The textures employed in this study were generated using a procedural approach. The color texture of snake skin is currently only capable of representing the body part texture with a more regular shape. The head and tail, which possess a more intricate color structure, or certain specialized body parts (e.g., the neck of the cobra), are not yet supported. In the future, it may be possible to model the patterns of these more complex parts and generate the associated textures.

Bibliography

[1] R. Bauchot, *Snakes: A Natural History*. Sterling Publishing Company, Incorporated, 2006, ISBN: 9781402731815. [Online]. Available: https://books.google.se/books?id=_RU3TQCH1esC.

[2] B. C. Hermawan, *Northern white-lipped python (leiopython albertisii)*, Feb. 2020. [Online]. Available: https://www.inaturalist.org/observations/39312374.

[3] M. E. McNamara, P. J. Orr, S. L. Kearns, L. Alcalá, P. Anadón, and E. Peñalver, "Reconstructing carotenoid-based and structural coloration in fossil skin," *Current Biology*, vol. 26, no. 8, pp. 1075–1082, 2016, ISSN: 0960-9822. DOI: https://doi.org/10.1016/j.cub.2016.02.038. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0960982216301208.

[4] J. Griffe, D. Bielsa, A. Jarabo, and A. Muñoz, "A biologically-inspired appearance model for snake skin," *Jornada de Jóvenes Investigadores del I3A*, vol. 11, Jul. 2023. DOI: 10.26754/jjii3a.20239057.

[5] A. Weidlich and A. Wilkie, "Arbitrarily layered micro-facet surfaces," in *Proceedings of the 5th International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia*, ser. GRAPHITE '07, Perth, Australia: Association for Computing Machinery, 2007, pp. 171–178, ISBN: 9781595939128. DOI: 10.1145/1321261.1321292. [Online]. Available: https://doi.org/10.1145/1321261.1321292.

[6] J. M. Pinheiro and M. Walter, "A procedural model for snake skin texture generation," in *VISIGRAPP*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:4370383.

[7] C. Donner and H. W. Jensen, "Light diffusion in multi-layered translucent materials," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1032–1039, Jul. 2005, ISSN: 0730-0301. DOI: 10.1145/1073204.1073308. [Online]. Available: https://doi.org/10.1145/1073204.1073308.

[8] C. Donner, T. Weyrich, E. d'Eon, R. Ramamoorthi, and S. Rusinkiewicz, "A layered, heterogeneous reflectance model for acquiring and rendering human skin," *ACM Trans. Graph.*, vol. 27, no. 5, Dec. 2008, ISSN: 0730-0301. DOI: 10.1145/1409060.1409093. [Online]. Available: https://doi.org/10.1145/1409060.1409093.

[9] J. Iglesias Guitián, C. Aliaga, A. Jarabo, and D. Gutiérrez, "A biophysically-based model of the optical properties of skin aging," *Computer Graphics Forum*, vol. 34, pp. 45–55, May 2015. DOI: 10.1111/cgf.12540.

[10] W. Huang, M. B. Hullin, and J. Hanika, "A microfacet-based hair scattering model," *Computer Graphics Forum*, vol. 41, no. 4, pp. 79–91, 2022. DOI: https://doi.org/10.1111/cgf.14588. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14588. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14588.

[11] S. R. Marschner, H. W. Jensen, M. Cammarano, S. Worley, and P. Hanrahan, "Light scattering from human hair fibers," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 780–791, Jul. 2003, ISSN: 0730-0301. DOI: 10.1145/882262.882345. [Online]. Available: https://doi.org/10.1145/882262.882345.

[12] L.-Q. Yan, C.-W. Tseng, H. W. Jensen, and R. Ramamoorthi, "Physically-accurate fur reflectance: Modeling, measurement and rendering," *ACM Trans. Graph.*, vol. 34, no. 6, Nov. 2015, ISSN: 0730-0301. DOI: 10.1145/2816795.2818080. [Online]. Available: https://doi.org/10.1145/2816795.2818080.

[13] M. Shawkey and L. D'Alba, "Interactions between colour-producing mechanisms and their effects on the integumentary colour palette," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 372, p. 20 160 536, May 2017. DOI: 10.1098/rstb.2016.0536.

[14] B. E. Smits and G. W. Meyer, "Newton's colors: Simulating interference phenomena in realistic image synthesis," in *Photorealism in Computer Graphics*, K. Bouatouch and C. Bouville, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 185–194. DOI: 10.1007/978-3-662-09287-3_13. [Online]. Available: https://doi.org/10.1007/978-3-662-09287-3_13.

[15] D. S. Dhillon, J. Teyssier, M. Single, I. Gaponenko, M. Milinkovitch, and M. Zwicker, "Interactive Diffraction from Biological Nanostructures," in *Eurographics 2014 - Posters*, M. Paulin and C. Dachsbacher, Eds., The Eurographics Association, 2014. DOI: 10.2312/egp.20141064.

[16] P. Hanrahan and W. Krueger, "Reflection from layered surfaces due to subsurface scattering," in *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '93, Anaheim, CA: Association for Computing Machinery, 1993, pp. 165–174, ISBN: 0897916018. DOI: 10.1145/166117.166139. [Online]. Available: https://doi.org/10.1145/166117.166139.

[17] Y. Guo, M. Haan, and S. Zhao, "Position-free monte carlo simulation for arbitrary layered bsdfs," *ACM Trans. Graph.*, vol. 37, no. 6, Dec. 2018, ISSN: 0730-0301. DOI: 10.1145/3272127.3275053. [Online]. Available: https://doi.org/10.1145/3272127.3275053.

[18] G. Cocho, R. Pérez-Pascual, and J. Rius, "Discrete systems, cell-cell interactions and color pattern of animals. i. conflicting dynamics and pattern formation," *Journal of Theoretical Biology*, vol. 125, no. 4, pp. 419–435, 1987, ISSN: 0022-5193. DOI: https://doi.org/10.1016/S0022-5193(87)80211-4. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022519387802114.

[19] J. G. Just, *Brazilian green racer (philodryas aestiva)*, Nov. 2018. [Online]. Available: https://www.inaturalist.org/observations/18583799.

[20] Snevares, *Dumerils diadem snake (phalotris lemniscatus)*, Feb. 2024. [Online]. Available: https://www.inaturalist.org/observations/200662083.

[21] P. E. Nahuat-Cervera, *Variable coralsnake (micrurus diastema)*, Oct. 2020. [Online]. Available: https://www.inaturalist.org/observations/61991728.

[22] M. Di-Bernardo, *Ptychophis flavovirgatus*. [Online]. Available: https://reptile-database.reptarium.cz/species?genus=Ptychophis&species=flavovirgatus.

[23] L. F. C. d. Lima, *Amazon false coral snake (oxyrhopus rhombifer)*, Dec. 2021. [Online]. Available: https://www.inaturalist.org/observations/102814875.

[24] E. B. Ong, *Ball python (python regius)*, Sep. 2023. [Online]. Available: https://www.inaturalist.org/observations/184843509.

[25] M. Pharr, W. Jakob, and G. Humphreys, *Physically based rendering: From theory to implementation*. The MIT Press, 2023.

[26] J. T. Kajiya, "The rendering equation," *SIGGRAPH Comput. Graph.*, vol. 20, no. 4, pp. 143–150, Aug. 1986, ISSN: 0097-8930. DOI: 10.1145/15886.15902. [Online]. Available: https://doi.org/10.1145/15886.15902.

[27] B. Walter, S. R. Marschner, H. Li, and K. E. Torrance, "Microfacet models for refraction through rough surfaces," in *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, ser. EGSR'07, Grenoble, France: Eurographics Association, 2007, pp. 195–206, ISBN: 9783905673524.

[28] C. Schlick, "An inexpensive bdrf model," Apr. 2002.

[29] T. Akenine-Mller, E. Haines, and N. Hoffman, *Real-Time Rendering, Fourth Edition*, 4th. USA: A. K. Peters, Ltd., 2018, ISBN: 0134997832.

[30] S. Eichelbaum, M. Hlawitschka, B. Hamann, and G. Scheuermann, "Image-space tensor field visualization using a lic-like method," Jan. 2012, pp. 193–210, ISBN: 978-3-642-21607-7. DOI: 10.1007/978-3-642-21608-4_11.

[31] G. Turk, "Generating textures on arbitrary surfaces using reaction-diffusion," *SIGGRAPH Comput. Graph.*, vol. 25, no. 4, pp. 289–298, Jul. 1991, ISSN: 0097-8930. DOI: 10.1145/127719.122749. [Online]. Available: https://doi.org/10.1145/127719.122749.

[32] A. Witkin and M. Kass, "Reaction-diffusion textures," *SIGGRAPH Comput. Graph.*, vol. 25, no. 4, pp. 299–308, Jul. 1991, ISSN: 0097-8930. DOI: 10.1145/127719.122750. [Online]. Available: https://doi.org/10.1145/127719.122750.

[33] M. McKay, *Perlin noise maker*. [Online]. Available: http://kitfox.com/projects/perlinNoiseMaker/.

[34] S. Worley, "A cellular texture basis function," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '96, New York, NY, USA: Association for Computing Machinery, 1996, pp. 291–294, ISBN: 0897917464. DOI: 10.1145/237170.237267. [Online]. Available: https://doi.org/10.1145/237170.237267.

[35] A. Doull, *Hello worley*, May 2011. [Online]. Available: https://procworld.blogspot.com/2011/05/hello-worley.html.

[36] P. Ivo, *Cobra-dágua (erythrolamprus miliaris)*, Mar. 2017. [Online]. Available: https://www.biodiversity4all.org/observations/5408315.

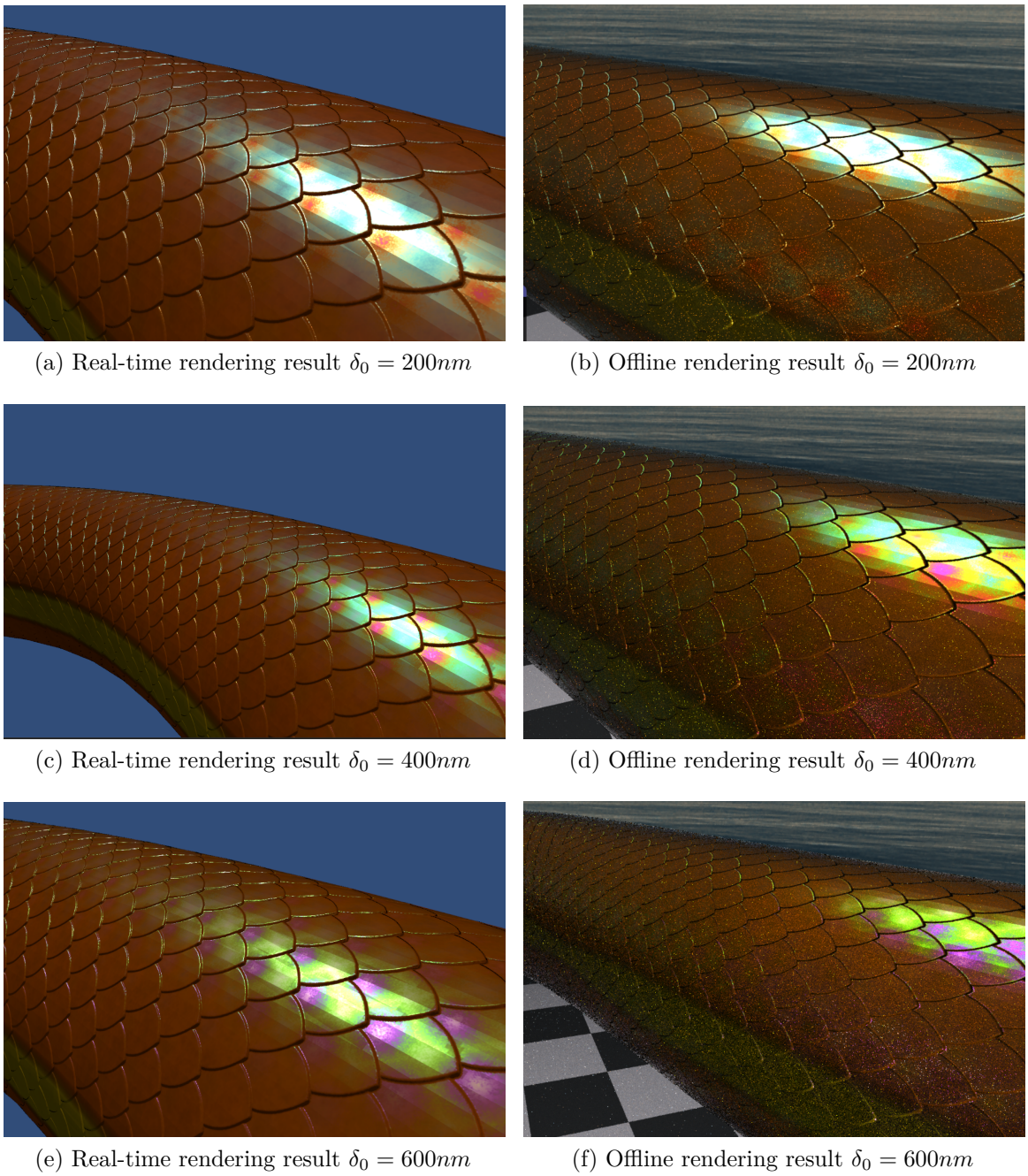
[37] J. Miller-Camp, *Eastern milksnake (lampropeltis triangulum)*, Sep. 2023. [Online]. Available: https://www.inaturalist.org/observations/184453331.

[38] S. G. B. Silveira, *Lichtensteins green racer (philodryas olfersii)*, Feb. 2024. [Online]. Available: https://www.inaturalist.org/observations/200429752.

[39] L. Belcour and P. Barla, "A practical extension to microfacet theory for the modeling of varying iridescence," *ACM Trans. Graph.*, vol. 36, no. 4, Jul. 2017, ISSN: 0730-0301. DOI: 10.1145/3072959.3073620. [Online]. Available: https://doi.org/10.1145/3072959.3073620.

A

Appendix 1



(a) Real-time rendering result $\delta_0 = 200nm$

(b) Offline rendering result $\delta_0 = 200nm$

(c) Real-time rendering result $\delta_0 = 400nm$

(d) Offline rendering result $\delta_0 = 400nm$

(e) Real-time rendering result $\delta_0 = 600nm$

(f) Offline rendering result $\delta_0 = 600nm$

Figure A.1: Comparison between real-time rendering and offline rendering with different thickness of the iridescent layer.