Final Project Report (Computer Graphics, COMP4610/8610-2025)

Jiaxin Xie	Jinghang Li	Junhao Liu	Qingchuan Rui
u8153316	u8162481	u7778766	u7776331

The Australian National University

Abstract

This project presents a physically based rendering system designed to replicate the award-winning image from the 2016 Dartmouth Rendering Competition by Srinath Ravichandran. Our implementation builds a Monte Carlo path tracer supporting realistic rendering of both rough and smooth conductor and dielectric materials using a microfacet Bidirectional Scattering Distribution Function (BSDF). We extend the renderer to support advanced effects such as field depth, using the Gaussian thin lens model, and chromatic dispersion. The final scene—featuring a high-resolution gold king chess piece and two rows of black pawns on a reflective chessboard—demonstrates complex light transport under area lighting at 2048 samples per pixel. Comparative analysis with Mitsuba validates the physical accuracy of our system. Additionally, a JSON-based interface allows for interactive scene configuration. This project demonstrates an integrated pipeline for cinematic quality rendering and material modeling grounded in modern graphics research.

1 Introduction

Photorealistic rendering is a central pursuit in computer graphics, aiming to simulate the complex interplay of light and materials as it occurs in the real world. This project seeks to reproduce compelling visual results through an implementation of a Monte Carlo path tracer augmented with microfacet BSDFs and physically accurate light transport models.

2 Motivation and Problem Statement

Our target scene—a cinematic composition of a reflective gold king chess piece under a soft area light, flanked by two rows of polished black pawns—presents a range of visual challenges. These include rendering metallic and dielectric materials with varying surface roughness, simulating photographic depth of field(DOF), capturing high-frequency specular highlights, and modeling light dispersion through transparent media. To meet these goals, we designed and implemented a rendering engine that includes: (1) a Monte Carlo Path Tracer with microfacet BSDF pipeline for conductors and dielectrics; (2) support for cinematic rendering effects such as focus and dispersion; (3) an interactive JSON interface for scene manipulation; and (4) a comparative validation of our results against the Mitsuba reference renderer.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

3 Previous Works

3.1 Path Tracing and Global Illumination

A variety of path tracing algorithms have been proposed for rendering realistic images, including bidirectional path tracing (8), Metropolis light transport (12), and photon mapping (2). Our project adopts the Monte Carlo path tracing approach (6) for its conceptual simplicity, extensibility, and compatibility with physically based rendering pipelines. Despite its higher variance compared to the aforementioned techniques, its simple, unbiased formulation makes it well-suited for our general-purpose rendering use case and offline production.

3.2 Microfacet BSDF Models

Accurately modeling surface reflectance is essential for photorealism. While Lambertian and Phong (9) models provide computationally cheap approximations, they lack physical justification. Microfacet theory (3) models surfaces as distributions of microscale facets, allowing for physically based treatment of both rough and smooth materials. We re-implement the BSDF model introduced by Walter et al. (13), which simulates reflection and refraction at rough boundaries between media utilizing the GGX microfacet distribution. This model effectively captures specular highlights and grazing-angle behavior, making it suitable for rendering conductor and dielectric materials.

3.3 DOF and Camera Models

Unlike pinhole cameras, which produce images where all scene objects are in sharp focus, real-world cameras use lens systems to achieve selective focus and aesthetic defocus effects. Simulating DOF enhances photographic realism and focus guidance. A standard method uses the thin lens approximation (7), where rays are sampled through a user-defined lens aperture and directed toward a point on the focal plane to produce defocus blur (4). We implement this model to recreate the cinematic focus seen in our reference image, highlighting the king-chess piece.

3.4 Chromatic Dispersion and Spectral Rendering

Spectral rendering extends traditional RGB rendering by modeling wavelength-dependent behavior. Dispersion effects, such as those seen in gemstones, occur as a result of wavelength-dependent refractive indices. The physical basis of light dispersion is that materials with non-constant refractive indices split white light into spectral components under Snell's law (11)(14). Sun et al. introduced a composite spectral model that extends ray tracers to decompose light into smooth and spiked spectral components, enabling the spawning of monochromatic rays based on Cauchy's dispersion equation (1). Their method achieves high visual fidelity with low implementation complexity, offering a practical approach to render gem-like caustics and spectral separation in transparent materials—capabilities critical for reproducing realistic dispersive effects.

4 Our Method

4.1 Path Tracing and Global Illumination

Monte Carlo path tracing is used for calculating the rendering equation(6):

$$
\begin{aligned}
L(\mathbf{x}, \omega_o, \lambda) &= L_e(\mathbf{x}, \omega_o, \lambda) + L_r(\mathbf{x}, \omega_o, \lambda) \\
L_i(\mathbf{x}, \omega_o, \lambda) &= \int_{\pm\Omega - A_l} f_s(\mathbf{x}, \omega_i, \omega_o, \lambda) L(\mathbf{x}, \omega_i, \lambda)(\omega_i \cdot \mathbf{n}) \mathrm{d}\omega_i \\
L_e(\mathbf{x}, \omega_o, \lambda) &= \int_{A_l} f_s(\mathbf{x}, \omega_l, \omega_o, \lambda) \frac{E(\mathbf{p}_l, \lambda)(-\omega_l \cdot \mathbf{n}_l)}{\|\mathbf{p}_l - \mathbf{x}\|^2} (\omega_l \cdot \mathbf{n}) \mathrm{d}A
\end{aligned}
\tag{1}
$$

Where L is the total spectral radiance of wavelength λ at position $\mathbf{x}$ and outward direction ω_o; L_e is the directly emitted radiance, L_i is the indirectly one; $\int_{\pm\Omega - A_l} \mathrm{d}\omega_i$ is an integral over the upper and lower unit hemisphere centred around the normal vector $\mathbf{n}$, without the directly emitted part; f_s is the bidirectional scattering distribution function (BSDF). For the directly emitted part, $\int_{A_l} \mathrm{d}A$

is an integral over all area lights; $E(\mathbf{p}_l, \lambda)$ is the spectral emitted irradiance of the light source at position $\mathbf{p}_l$ and wavelength λ; ω_l is the direction to the light sources; $\mathbf{n}_l$ is the normal vector of the light sources.

The equation is highly based on the physics of radiometry, and calculating it gives us a physically based rendering with realistic global illumination.

The Monte Carlo method is used to approximate the integral parts of 1.

$$\int_{\infty} g(x)\mathrm{d}x \approx \frac{1}{N}\sum_{i=1}^{N}\frac{g(X_i)}{p(X_i)} \tag{2}$$

where $X_i \sim p(x)$ and $p(x)$ is a probability distribution function (PDF).

Sampling points form paths rather than trees to avoid the exponential calculation explosion by setting $N = 1$ in 2. Russian roulette is needed as a termination strategy since 1 is in a recursive form. We approximate the equation under a Bernoulli random variable $R \sim b(1, p_{rr})$. Then replacing $g(X_i)$ with $R_i g(X_i)/p_{rr}$ gives an unbiased approximation:

$$E[Rg(X)/p_{rr}] = p_{rr} \cdot \frac{E[g(X)]}{p_{rr}} = E[g(X)] \tag{3}$$

Combining the three equations, we can get the final equation used in the Monte Carlo path tracing:

$$L(\mathbf{x}, \omega_o, \lambda) \approx L_e(\mathbf{x}, \omega_o, \lambda) + R\frac{f_s(\mathbf{x}, \omega_i, \omega_o, \lambda)L(\mathbf{x}, \omega_i, \lambda)(\omega_i \cdot \mathbf{n})}{p(\omega_i; \omega_o, \lambda)p_{rr}} \tag{4}$$

4.2 Microfacet BSDF Models

One of the most important parts in 1 is the BSDF term, which depicts the material properties at the position $\mathbf{x}$. We used a microfacet-based BSDF model for the term, where we see a rough surface as many randomly distributed micro mirrors, which can be described as a random normal vector $\mathbf{h} \sim D(\mathbf{h}; \mathbf{n})$.

For reflectance, the BRDF is given by the Torrance–Sparrow BRDF model(3):

$$f_r(\omega_i, \omega_o, \lambda) = \frac{F(\omega_i, \mathbf{h}, \lambda)D(\mathbf{h})G(\omega_i, \omega_o, \mathbf{n})}{4(\omega_i \cdot \mathbf{n})(\omega_o \cdot \mathbf{n})} \tag{5}$$

Where $F(\omega_i, \mathbf{h}, \lambda)$ is the Fresnel coefficient; $D(\mathbf{h})$ is the distribution of $\mathbf{h}$; $G(\omega_i, \omega_o, \mathbf{n})$ is the shadow-masking term that consider the possibilities that light or view is blocked by other micro surfaces.

And since now we are sampling $\mathbf{h}$, the PDF in 4 should be calculated based on $D(\mathbf{h})$:

$$\begin{aligned} \mathbf{h} &= \frac{\omega_i + \omega_o}{\|\omega_i + \omega_o\|} \Rightarrow \left\|\frac{\partial \mathbf{h}}{\partial \omega_i}\right\| = \frac{1}{4(\omega_o \cdot \mathbf{h})} \\ p(\omega_i; \omega_o, \lambda) &= p_h(\mathbf{h})\left\|\frac{\partial \mathbf{h}}{\partial \omega_i}\right\| = \frac{D(\mathbf{h})(\mathbf{n} \cdot \mathbf{h})}{4(\omega_o \cdot \mathbf{h})} \end{aligned} \tag{6}$$

A similar technique is applied for the refraction, but with a more complex form (13). Consider the outside-in refraction:

$$f_t(\omega_i, \omega_o, \lambda) = \frac{\eta^2(\lambda)(1 - F(\omega_i, \mathbf{h}, \lambda))D(\mathbf{h})G(\omega_i, \omega_o, \mathbf{n})}{((\omega_i \cdot \mathbf{h}) + \eta(\lambda)(\omega_o \cdot \mathbf{h}))^2}\frac{|\omega_i \cdot \mathbf{h}||\omega_o \cdot \mathbf{h}|}{|\omega_i \cdot \mathbf{n}||\omega_o \cdot \mathbf{n}|} \tag{7}$$

where $\eta(\lambda) = n_o/n_i$ is the refractive index.

For the PDF:

$$\mathbf{h} = \frac{-\omega_i - \eta\omega_o}{\|\omega_i + \eta\omega_o\|} \Rightarrow \left\|\frac{\partial \mathbf{h}}{\partial \omega_i}\right\| = \frac{\eta^2|\omega_o \cdot \mathbf{h}|}{((\omega_i \cdot \mathbf{h}) + \eta(\omega_o \cdot \mathbf{h}))^2} \tag{8}$$

For the D and G term, we used the GGX distribution for more efficient computation compared to other choices with exponential terms (13):

$$\begin{aligned} D(\mathbf{h};\alpha) &= \frac{\alpha^2}{\pi\cos^4\theta(\alpha^2+\tan^2\theta)^2} \\ G(\omega_i,\omega_o,\mathbf{n}) &= G_1(\omega_i,\mathbf{n})G_1(\omega_o,\mathbf{n}) \\ G_1(\omega,\mathbf{n};\alpha) &= \frac{2}{1+\sqrt{1+\alpha^2\tan^2\theta}} \end{aligned} \tag{9}$$

Where θ is the polar angle with the original normal vector; α is a parameter that controls the roughness of a material.

By decoupling sampling and evaluation, importance sampling directs more rays toward microfacet orientations that contribute the most to the specular component, thereby significantly reducing variance in the rendered image.

4.3 DOF and Camera Models

We employ a thin lens camera model based on the Gaussian equation:

$$\frac{1}{d_o}+\frac{1}{d_i}=\frac{1}{f} \tag{10}$$

where d_o is the object distance, d_i is the image (sensor) distance, and f is the focal length. Given a pinhole ray direction $\mathbf{d}$ from the lens centre $\mathbf{o}$, the intersection point on the focal plane is computed as:

$$\mathbf{p}_f = \mathbf{o} + \frac{d_i}{d_z}\mathbf{d} \tag{11}$$

where d_z is the z-component of $\mathbf{d}$ in camera space. To construct the final DOF ray, we sample a point $\mathbf{p}_l$ on the lens (according to the aperture radius) and compute the ray direction toward the focal point $\mathbf{p}_f$:

$$\text{origin} = \mathbf{p}_l, \quad \text{direction} = \frac{\mathbf{p}_f - \mathbf{p}_l}{\|\mathbf{p}_f - \mathbf{p}_l\|} \tag{12}$$

This formulation enables physically accurate DOF rendering, producing sharp focus at the specified distance and bokeh for defocused regions.

4.4 Chromatic Dispersion and Spectral Rendering

We implemented the spectral rendering by assigning wavelengths to the ray and calculating the R, G, and B channels for different wavelengths. The wavelengths for the three channels are the standard wavelengths used in the CIE 1931 color space standard (10), which are:

$$\lambda_R = 700\text{nm} \qquad \lambda_G = 546.1\text{nm} \qquad \lambda_B = 435.8\text{nm}$$

The main effect from spectral rendering is the chromatic dispersion, which is due to the change of refractive index at different wavelengths. Cauchy's equation can depict such a change nicely (1):

$$\eta(\lambda) = A_0 + \sum_{i=1}\frac{A_i}{\lambda^{2i}} \tag{13}$$

Our implementation obtained a decent result despite only considering the first two terms, A_0 and A_1, for simplicity.

5 Algorithms and Design

5.1 Path Tracing and Global Illumination

Our project builds on top of the code framework provided by the University of California, Santa Barbara GAMES101 assignment (15), where we added modifications to implement our Monte

Carlo Path Tracer with BVH acceleration. This section details the unbiased Monte Carlo path tracing algorithm(6) used to simulate global illumination, drawing directly on our implementation in Scene::castRay and Renderer::Render.

Algorithm 1 Path Tracing (per pixel)

Input : Number of samples per pixel *spp*
Output : Rendered image buffer image

for *each pixel* p **do**
  $L \leftarrow \mathbf{0}$ **for** $s \leftarrow 1$ **to** *spp* **do**
    ray $\leftarrow$ GenerateCameraRay(p) $L \leftarrow L +$ TracePath(ray, 0)
  image[p] $\leftarrow L/spp$

Algorithm 2 Function TracePath(ray, depth)

Function `TracePath` *(ray, depth)***:**
  inter $\leftarrow$ Scene.intersect(ray) **if** *not* inter.happened **then**
    if useEnvMap **then**
      return SampleEnv(ray.direction)
    else
      return backgroundColor
  if $depth = 0$ ***and*** inter.obj.isEmitter **then**
    return inter.m.emission $\times$ $|\cos\theta|$
  $L_{\text{dir}} \leftarrow$ FresnelWeight $\times$ Scene.directLighting(...) **if** random() $\geq$ rrRate **then**
    return L_{dir}
  $(\omega_i, \mathit{isReflect}) \leftarrow$ inter.m.sampleOrRefract(...)
  pdf $\leftarrow$ inter.m.pdf(...)
  $L_{\text{ind}} \leftarrow$TracePath$\big($Ray(inter.p $+ \varepsilon\,\mathbf{n}, \omega_i)$, $depth + 1\big) \times$ inter.m.eval(...) $\times$ $|\cos\theta|$
      / pdf / rrRate
  return clamp($L_{\text{dir}} + L_{\text{ind}}$)

Step 1 shoots the ray into the BVH-accelerated scene. If no geometry is hit, an environment map or background color is returned. Upon hitting an emitter and at depth 0, direct emission is accounted for. Direct lighting uses multiple importance sampling over area lights, accumulating contributions where unoccluded.

Russian Roulette probabilistically terminates paths to bound recursion cost while maintaining unbiasedness. For surviving paths, we sample either microfacet-based reflection or refraction via Material::sample and compute the corresponding BSDF value and PDF. A recursive call evaluates incoming radiance from direction wi, scaled by BSDF response, geometry term, PDF, and RR weighting. The sum of direct and indirect components, clamped to mitigate fireflies, yields the pixel's radiance.

This algorithm robustly captures both local and global illumination, supporting phenomena from soft shadows and caustics to complex multi-bounce color bleeding.

5.2 Microfacet BSDF

The microfacet BSDF implementation in `material.hpp` follows the physically based model (3), leveraging the GGX distribution for surface normal microfacet orientation and the Smith masking–shadowing function for visibility.

The microfacet BSDF pipeline samples a half-vector $\mathbf{h}$ from the GGX normal distribution function parameterised by surface roughness α, where α corresponds to the `material.roughness` parameter. Depending on the material type, the sampled half-vector is used to compute either a reflection or a refraction direction: conductors employ perfect mirror reflection, whereas dielectrics apply Snell's law with relative index η. We sample outgoing direction $\mathbf{w}_i$ using the PDF described

in 8 for unbiased Monte Carlo integration. The BSDF value $f(\mathbf{l}, \mathbf{v})$ is evaluated by combining the normal distribution function D, the Smith masking–shadowing term G, and the Fresnel factor F, normalized by $4\,(\mathbf{n}\cdot\mathbf{l})\,(\mathbf{n}\cdot\mathbf{v})$.

Algorithm 3 Microfacet Sampling (`Material::sample`)

Input: random samples u_1, u_2, view direction $\mathbf{v}$, roughness α
Output: sampled direction $\mathbf{w}_i$, PDF p
$\mathbf{h} \leftarrow \mathrm{SampleGGX}(u_1, u_2, \alpha)$
if *isDielectric* **then**
  $\mathbf{w}_i \leftarrow \mathrm{Refract}(\mathbf{v}, \mathbf{h}, \eta)$
else
  $\mathbf{w}_i \leftarrow \mathrm{Reflect}(\mathbf{v}, \mathbf{h})$
$p \leftarrow \dfrac{D(\mathbf{h})\,|\mathbf{n}\cdot\mathbf{h}|}{4\,|\mathbf{v}\cdot\mathbf{h}|}$ **return** $\mathbf{w}_i$, p

Algorithm 4 Microfacet Evaluation (`Material::eval`)

Input: incoming $\mathbf{l}$, outgoing $\mathbf{v}$, roughness α, base Fresnel F_0
Output: BSDF value $f(\mathbf{l}, \mathbf{v})$
$\mathbf{h} \leftarrow \mathrm{normalize}(\mathbf{l} + \mathbf{v})$
$D \leftarrow \mathrm{GGX\_D}(\mathbf{h}, \alpha)$ $G \leftarrow \mathrm{SmithG}(\mathbf{l}, \mathbf{v}, \mathbf{h}, \alpha)$ $F \leftarrow \mathrm{Fresnel\_Schlick}(\mathbf{v}\cdot\mathbf{h}, F_0)$
return $\dfrac{D\,G\,F}{4\,|\mathbf{n}\cdot\mathbf{l}|\,|\mathbf{n}\cdot\mathbf{v}|}$

5.3 Camera Ray Generation with DOF

Our implementation of DOF occurs within `renderer.cpp`, where each camera ray is generated using a thin-lens approximation with randomised aperture sampling and a focus-driven ray redirection. This algorithm details the per-pixel ray construction process.

Algorithm 5 Camera Ray Generation with Thin-Lens DOF

Input: Pixel (px, py); random jitters u_1, u_2, u_3, u_4; lens radius r_{lens}; focal distance f
Output: Ray with lens offset and direction toward focal plane
$P \leftarrow \mathit{lowerLeft} + \dfrac{px + u_1}{\mathit{width}} \cdot \mathit{horizontal} + \dfrac{py + u_2}{\mathit{height}} \cdot \mathit{vertical}$
$\mathbf{d} \leftarrow \mathrm{normalize}(P - \mathit{origin})$
$\mathbf{s} \leftarrow \mathrm{ConcentricSampleDisk}(u_3, u_4) \cdot r_{\text{lens}}$
$\mathbf{o}_{\text{offset}} \leftarrow \mathbf{u} \cdot s_x + \mathbf{v} \cdot s_y$
$t_{\text{focus}} \leftarrow \dfrac{f}{\mathbf{d} \cdot \mathbf{w}}$
$P_{\text{focus}} \leftarrow \mathit{origin} + \mathbf{d} \cdot t_{\text{focus}}$
$\mathit{newOrigin} \leftarrow \mathit{origin} + \mathbf{o}_{\text{offset}}$
$\mathit{newDir} \leftarrow \mathrm{normalize}(P_{\text{focus}} - \mathit{newOrigin})$
return $\mathrm{Ray}(\mathit{newOrigin}, \mathit{newDir})$

The ray generation algorithm begins by mapping the pixel coordinates (with sub-pixel jitter) onto the sensor plane to obtain a nominal direction $\mathbf{d}$. A point on the lens is then randomly sampled using concentric disk mapping, scaled by r_{lens}, to simulate a finite aperture. The intersection point on the focal plane is determined by analytically solving for the distance along $\mathbf{d}$ required to reach the focal depth f. Finally, a ray is constructed from the shifted lens sample position toward this focal point. This strategy enables defocus blur and bokeh effects to emerge naturally from the ray sampling process in the Monte Carlo renderer.

5.4 Environment Map

We incorporate an environment map loaded and queried in `scene.hpp`/`scene.cpp`.

Algorithm 6 Scene::loadEnvMap(path)

Input : path
Output : envMap loaded into the scene

envMap ← ImageHDR(path) `// load HDR or LDR buffer`
width ← envMap.width() height ← envMap.height()

buildEnvCDF(envMap, cdfRows, cdfCols)

Algorithm 7 Scene::sampleEnv(dir) **const**

Input : dir
Output : Environment map texel corresponding to direction

$\theta \leftarrow \arccos\big(\text{clamp}(dir.y,\, -1,\, 1)\big)\;\; \phi \leftarrow \text{atan2}(dir.z,\, dir.x) + \pi$

$u \leftarrow \phi/(2\pi)\;\; v \leftarrow \theta/\pi$

$i \leftarrow \text{clamp}\big(\lfloor u \times width \rfloor,\, 0,\, width - 1\big)\;\; j \leftarrow \text{clamp}\big(\lfloor v \times height \rfloor,\, 0,\, height - 1\big)$

return envMap.texel(i, j)

Loading (`loadEnvMap`) reads an HDR/PNG file into a floating-point buffer and records its dimensions; cumulative distribution functions (*cdfRows*, *cdfCols*) may be precomputed for importance sampling bright regions. Spherical mapping (`sampleEnv`) converts a normalized direction **dir** to angles θ (polar) and ϕ (azimuthal), then scales them to UV coordinates for texture lookup. Texel retrieval returns the stored RGB radiance for rays that escape the scene.

Within `Scene::castRay`, environment mapping is invoked when no intersection occurs:

$$\texttt{if (!inter.happened)} \begin{cases} \text{sampleEnv}(ray.direction) \times EnvScale & ,\text{useEnv} \\ backgroundColor & ,\text{otherwise} \end{cases}$$

6 Experiments

6.1 Evaluation on Microfacet BSDF

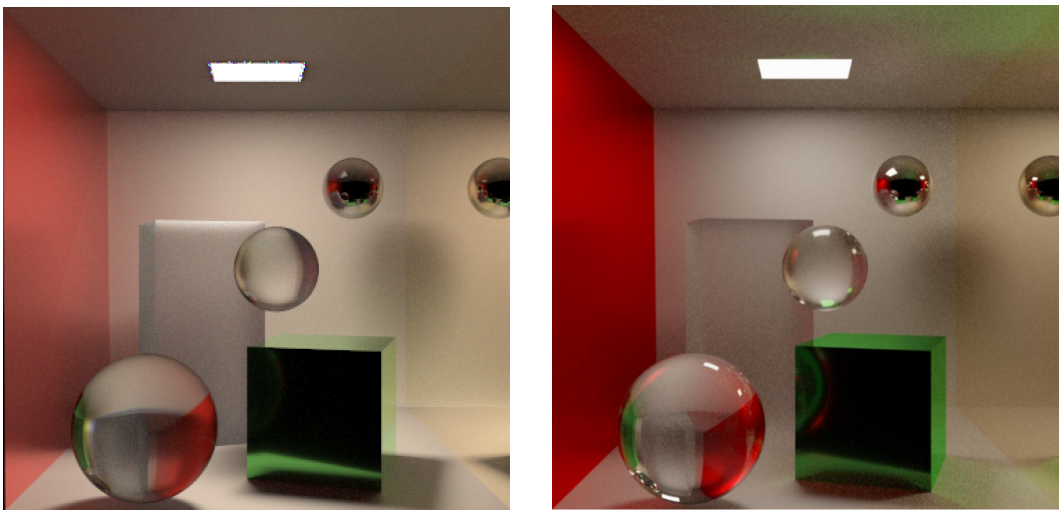
We evaluate the correctness of our microfacet BSDF implementation by rendering a Cornell box with eight materials, including both conductors and dielectrics at varying roughness levels. We perform a qualitative comparison with the reference image rendered using Mitsuba 3.

Smooth conductors, such as the silver sphere (top right) and the golden wall, demonstrate highly specular reflections, producing sharp, mirror-like images of the environment. Green cube with low roughness also reflects its surroundings crisply, indicating that the microfacets are tightly aligned, as expected for a nearly smooth conductor.

The smooth dielectric sphere (bottom left) shows clean internal reflections and highlights. The light bends through it predictably, and the reflections are still relatively sharp. In comparison, the rough dielectric sphere (center) preserves strong transmission but introduces a slight blurring of refraction, indicating appropriate microfacet-induced scattering.

The dielectric cuboid with higher roughness appears frosted and diffuse, exhibiting scattered transmission and significantly softened highlights. Similarly, the white conductor back wall appears matte and noisy, with minimal visible reflection, demonstrating that increased roughness has significantly broadened the specular lobe and reduced directional reflectivity.

Overall, the results show good optical behavior consistent with microfacet theory. However, qualitative inspection between our result and the Mitsuba result reveals that our outputs exhibit noticeably



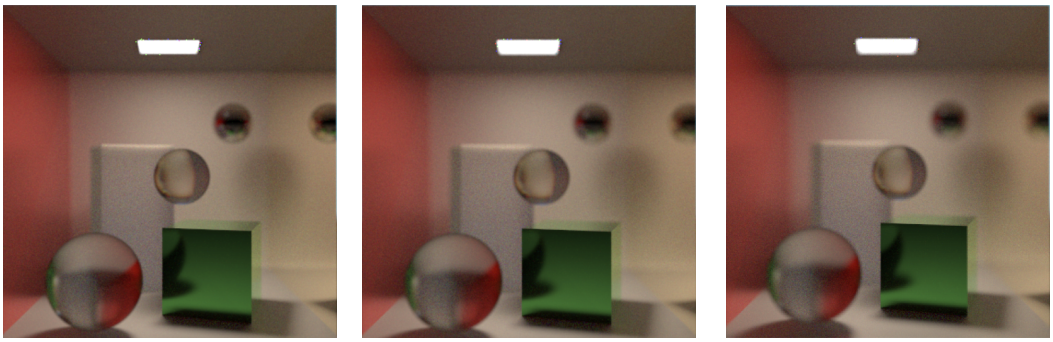
(a) Our rendered Cornell box (left)

(b) Mitsuba 3 reference render (right)

Figure 1: Cornell box scene demonstrating a variety of microfacet BSDF materials. Smooth conductors: small silver sphere (top right), golden right wall. Smooth dielectric: large sphere (bottom left). Rough conductors: green cube ($\alpha = 0.01$), red left wall ($\alpha = 0.1$), and white back wall ($\alpha = 0.4$). Rough dielectrics: medium floating sphere ($\alpha = 0.01$) and cuboid ($\alpha = 0.5$).

darker and less specular reflected surfaces. We hypothesize that this discrepancy arises from the characteristics of the GGX normal distribution function employed in our microfacet BSDF implementation. Unlike the Phong or Beckmann distributions, GGX has a heavier tail, which increases the probability of sampling microfacet normals at grazing angles. Consequently, the Smith masking–shadowing term experiences more frequent self-occlusion events, attenuating the reflected radiance and producing the observed dimming effect.

6.2 Evaluation on DOF



(a) Aperture radius = 20

(b) Aperture radius = 30

(c) Aperture radius = 40

Figure 2: DOF comparison with increasing aperture radius.

To validate our DOF implementation, we set the camera to focus on the green cube in the scene. The focal distance was computed as the scalar projection of the vector from the camera position $\mathbf{C}$ to the cube position $\mathbf{P}$ onto the normalized view direction $\mathbf{D}$, using the formula:

$$\text{Focal distance} = (\mathbf{P} - \mathbf{C}) \cdot \frac{\mathbf{T} - \mathbf{C}}{\|\mathbf{T} - \mathbf{C}\|} \tag{14}$$

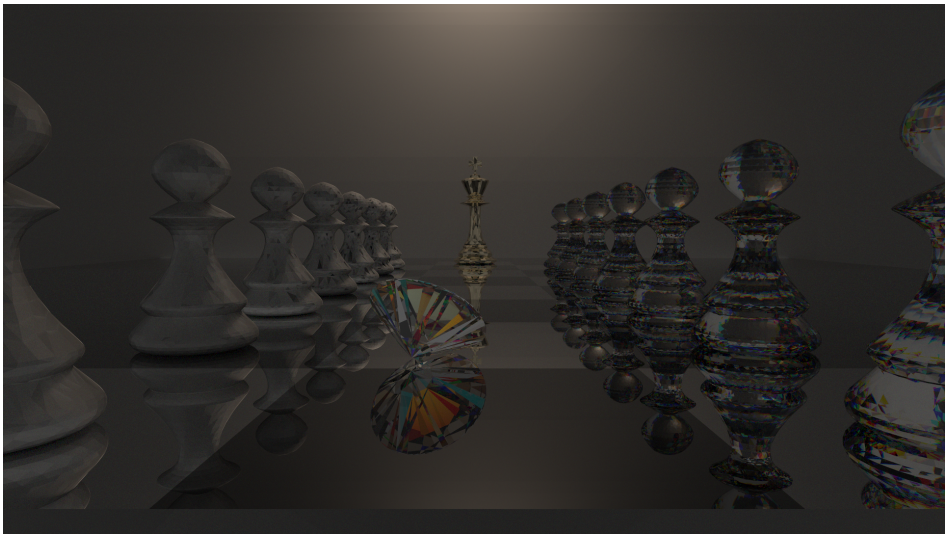
where $\mathbf{T}$ is the camera's look-at point. This approximation effectively sets the focal plane to pass through the green cube.

To evaluate the visual impact of the aperture size, we rendered three images with increasing aperture radii: 20, 30, and 40. As expected, a larger aperture produces a stronger blurring effect for objects that lie outside the focal plane, confirming the physical correctness of our lens simulation.

6.3 Final Rendering Result

We present two final renderings of a chessboard scene featuring identical king and pawn arrangements(5). The first rendering employs a sky environment map, simulates DOF effects with an aperture radius of 10, uses a field of view (FoV) of 75 degrees, and places an area light with intensity 100 at a height of 1300 units. The second rendering contrasts this by using a black background without DOF, a narrower FoV of 45 degrees, and a significantly lower light intensity of 1 at a height of 1600 units. Both images were rendered at 1920×1080 resolution, using 2024 samples per pixel, and 32 direct light samples. Rendering was performed on a CPU using an OpenMP-based 8-threaded parallel renderer, with each image completing in approximately 2 hours.

(a) Sky environment, DOF, FoV = 75, light intensity = 100



(b) Black background, no DOF, FoV = 25, light intensity = 1

Figure 3: Comparison of final chessboard scenes with identical object placement but different rendering configurations.

7 Discussions and Conclusion

In this work, we have developed a physically based Monte Carlo path tracer capable of reproducing complex light–material interactions in a cinematic chess scene. Our renderer integrates a microfacet BSDF framework, supporting both conductors and dielectrics via the GGX distribution and Smith masking–shadowing function—with depth-of-field effects modeled using a thin-lens approximation, and chromatic dispersion implemented through spectral sampling using standard CIE wavelengths. A JSON-driven interface facilitates interactive scene configuration, and comparisons against Mitsuba 3 validate our implementation's physical plausibility.

Limitations still remain: our renders exhibit a systematic reduction in brightness relative to Mitsuba's ground-truth outputs. We attribute this dimming to differences in the statistical characteristics of our sampling strategies: the GGX half-vector distribution possesses heavier tails than alternative microfacet models, which increases the probability of grazing-angle microfacet orientations, and it amplifies Smith masking–shadowing attenuation. Additionally, the lack of multiple importance sampling over the emitter's projected solid angle, potentially under-weighting high-contribution directions. Finally, minor discrepancies in Fresnel normalization and gamma correction between our implementation and Mitsuba's reference may further contribute to the observed luminance gap.

Looking ahead, we plan adding wavelength-dependent effects and more sophisticated light transport: first by modeling colored dielectrics using the Beer–Lambert law so that tinted absorption naturally blends with Fresnel reflections and refractions; next, introducing a subsurface-scattering module to capture the soft, diffusive appearance of materials like wax; then we move to per-ray spectral sampling to improve color accuracy beyond fixed RGB channels; and finally, integrate HDR environment-map importance sampling—alongside the existing area-light system—to deliver richer, scene-dependent illumination that better approximates real-world lighting.

These enhancements promise to close the remaining visual gap with reference renderers, broaden the range of supported materials, and elevate the rendering quality towards production-ready photorealism.

References

[1] Augustin-Louis Cauchy. *Sur la réfraction et la réflexion de la lumière*, page 151–157. Cambridge Library Collection - Mathematics. Cambridge University Press, 2009.

[2] Per Christensen, Henrik Wann Jensen, and Frank Suykens. A practical guide to global illumination using photon mapping. ACM SIGGRAPH 2001 Course Notes 38, August 2001.

[3] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Trans. Graph.*, 1(1):7–24, Jan. 1982.

[4] Joe Demers. Depth of field: A survey of techniques. *Gpu Gems*, 1(375):U390, 2004.

[5] dhruvin_doctor. chess 3d model, 2016.

[6] James T. Kajiya. The rendering equation. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '86, page 143–150, New York, NY, USA, 1986. Association for Computing Machinery.

[7] Craig Kolb, Don Mitchell, and Pat Hanrahan. A realistic camera model for computer graphics. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, page 317–324, New York, NY, USA, 1995. Association for Computing Machinery.

[8] Eric P. Lafortune and Yves D. Willems. Bi-directional path tracing. In *Proceedings of Third International Conference on Computational Graphics and Visualization Techniques (Compugraphics '93)*, pages 145–153, Alvor, Portugal, December 1993.

[9] Bui Tuong Phong. Illumination for computer generated pictures. *Commun. ACM*, 18(6):311–317, June 1975.

[10] T Smith and J Guild. The c.i.e. colorimetric standards and their use. *Transactions of the Optical Society*, 33(3):73, jan 1931.

[11] Yinlong Sun, F David Fracchia, and Mark S Drew. Rendering light dispersion with a composite spectral model. *Diamond*, 2(37.17):0–044, 2000.

[12] Eric Veach and Leonidas J. Guibas. Metropolis light transport. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, page 65–76, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[13] Bruce Walter, Stephen R. Marschner, Hongsong Li, and Kenneth E. Torrance. Microfacet models for refraction through rough surfaces. In *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, EGSR'07, page 195–206, Goslar, DEU, 2007. Eurographics Association.

[14] Wiseman. Final project - the perfect diamond cs348b - image synthesis, 2005.

[15] Lingqi Yan. Games: Graphics and mixed environment seminar games101 pa 6, 2020.

8 Peer Review

My name is JiaXin Xie, a COMP4610 student for 2025S1. Our project (titled: Monte Carlo Path Tracer with Microfacet BSDF) has been completed by a 4-member group consisting of JiaXin Xie, Jinghang Li, Junhao Liu, and Qingchuan Rui. From my best knowledge and assessment, l certify that the relative contributions among the team are:

Name	ID	Main duties and contributions	Contribution weighting
JiaXin Xie	u8153316	Discussion, main coder, and report writing.	25%
Jinghang Li	u8162481	Discussion, main coder, and report writing.	25%
Qingchuan Rui	u7776331	Scene construction, report writing.	25%
Junhao Liu	u7778766	Discussion, main coder, and report writing.	25%