Assignment 3: Basic Ray Tracing

NAME: YE HAOXU
STUDENT NUMBER: 2023533137
EMAIL:

1 INTRODUCTION

This report summarizes the implementation of a basic ray tracer for CS171 Assignment 3. The renderer supports BVH acceleration, analytic ray-triangle/AABB intersections, physically based BSDFs (Lambertian and perfect refraction), and both direct-lighting and random-walk style integration. Scenes are described by JSON configuration files and rendered to EXR images through the provided framework.

2 IMPLEMENTATION DETAILS

2.1 Acceleration and Intersections

- **BVH:** Implemented in `bvh_tree.h` / `bvh_accel.cpp`. Nodes are split by the median of centroids along the axis of maximum extent with a depth cap (20) and minimum leaf size (4 primitives).
- **Ray-Triangle:** `accel.cpp` uses the double-precision Möller-Trumbore algorithm. Ray time bounds are respected so secondary hits remain consistent with differential rays.
- **Ray-AABB:** Implemented via the slab method with precomputed inverse directions for robustness.

2.2 BSDFs and Integrators

- **Lambertian Diffuse:** Samples cosine-weighted directions and evaluates ρ/π.
- **Perfect Refraction:** Applies Snell's law through `Refract` with total-internal-reflection fallback to `Reflect`. Entering/exiting media are detected from $\mathbf{w}_o \cdot \mathbf{n}$.
- **IntersectionTestIntegrator:** Shoots rays until a diffuse hit is found, then performs point-light shading with a visibility test.
- **PathIntegrator:** Implements a classic path tracer with Russian roulette, throughput accumulation, BSDF sampling, and next-event estimation for emissive lights. Samples are averaged over `spp` per pixel.

2.3 Rendering Pipeline

Scenes in `data/` configure camera, materials, and integrator type. Rendering is parallelized with OpenMP. Output EXR files are tonemapped using `exrtools` for preview.

3 RESULTS

3.1 Tests

The following tests pass after the final implementation:

- Geometry/Math: `math_tests`, `intersection_tests`, `bvh_tests`, `kdtree_tests`.
- Texture/Property handling: `texture_tests`, `factory_tests`, `properties_tests`.
- Distribution/SD-tree infrastructure: `distribution_tests`, `sdtree_tests`.
- Full pipeline: `integration_tests` (all three scenes converge to the reference averages with 1×10^{-3} tolerance).

3.2 Renderings

Representative outputs (all available under `data/`):

- `cbox_no_light.exr`: diffuse Cornell box lit by a point source, showing soft shadows.
- `cbox_no_light_refract.exr`: demonstrates multi-bounce refraction through the glass sphere.
- `glass_sphere.exr`: validates that the path integrator preserves caustics under multiple scattering.

All renders exhibit stable noise patterns that diminish with increased `spp`. The light flux in the Cornell scenes was tuned to $[8.5, 6.0, 2.5]$ to match reference exposure.

4 CONCLUSION

The project delivers a fully functional HW3 renderer with both direct-lighting and path-tracing integrators, verified through the provided automated tests and sample images.