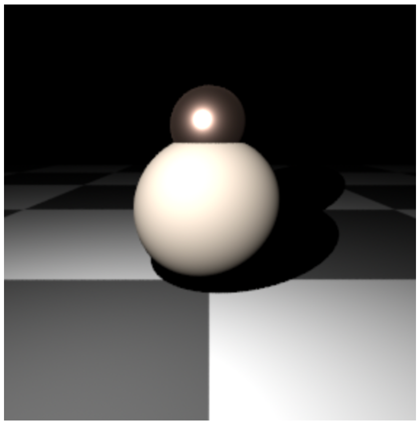
CSE 5545 Lab 3
Due: March 18 2024 11:59 PM
4 tasks, 12 points total

Objectives

- o Understand PBRT materials.
- o Understand accelerators in rendering.

Task 1: Create Composite Material (4 pts)

Continued from your lab 2, the first task is to extend your composite shade by adding suitable material properties. To make sure you understand how materials work in pbrt, you are asked to assign one unique material to each of the basic shapes in you composite shapes, i.e., creating a composite material. For example, below the snowman has *'metal'* as material for its head and has *'matte'* material for its body:



Hints:

- Snowman is shown here only as an example; you should create composite materials properly for your own composite shapes.

- You need to know how materials are applied to shapes (in pbrtShape() from api.cpp). One shape instance is bound to one material instance, together forming one primitive.

- Review your composite shape class implemented in lab 2. Especially in the Intersect() function. You need to nderstand the structure and what information are collected in SurfaceInteraction isect.

- Familiarize with the MixMaterial class in PBRT, which allows for a mixing of two materials based on a specified blend factor. The composite material class is inspired by MixMaterial class.

- Similar to shapes/snowman.cpp and shapes/snowman.h in the previous lab, create two new files: materials/compmat.cpp and materials/compmat.h.
    - Create new composite material class CompMaterial inherit from Material class:
        - Similar to the implementation in MixMaterial, you need to implement three functions for CompMaterial class:
            - the Constructor
            - the ComputeScatteringFunctions(). Fully understand MixMaterial::ComputeScatteringFunctions() will be helpful.
            - and CreateCompMaterial() which will be used in api.cpp for parsing.

- There are many ways to implement ComputeScatteringFunctions(). The key is to **differentiate each object in your composite shape so that you can assign different material to it**. One simple direction is to add an integer attribute Material_Id to SurfaceInteraction class. Another way is to add this attribute in the base class (such as Sphere class for snowman).

- The scene file for this might look like, for example:

```
MakeNamedMaterial "matteMaterial" "string type" "matte" "rgb Kd" [0.7 0.6 0.5]
MakeNamedMaterial "shinyMetal" "string type" "metal"

AttributeBegin
Material "comp" "string namedmaterial1" "matteMaterial" "string namedmaterial2" "shinyMetal"
Shape "snowman"    # snowman is my example, you should use your own shape
AttributeEnd
```

- Make sure to run cmake .. when you add new files into the codebase.

Task 2: Import one or multiple triangle meshes (2 pts)

- Now, enhance your scene by importing one or multiple triangle meshes of your choice to your current scene (the scene should include your composite shape/material as well as the triangular meshes).

- You should put effort to create a good scene (triangle mesh(es), shapes, etc. with proper material). Your grade will be based on your output picture quality. However, ensure the complexity is manageable to prevent excessive rendering times.

- Some resources:
    - https://benedikt-bitterli.me/resources/
    - https://www.artec3d.com/3d-models/ply

- Make sure to scale down the size of the mesh if it is too large for the view.

Task 3: Accelerator Comparison (3 pts)

The last task is to experiment with different acceleration structures. In PBRT, both BVH (Bounding Volume Hierarchy) and KD-tree (k-dimensional tree) can be used to improve the efficiency of ray tracing, so that the number of intersection tests during rendering is minimized.

Run experiments and report the performance of using accelerators for `coffee.pbrt` in this step.

- Take `coffee.pbrt` as the scene (found in lab3.zip), you will get the following image:

- Run the following experiments:
  - Modifying coffee.pbrt scene with a BVH accelerator. You are asked to experiment with four different splitting methods: SAH, HLBVH, Middle, and EqualCounts. Retain default settings for all other parameters.
  - Modify this coffee.pbrt scene with the KD-tree accelerator using its default settings.
- Gather results from the above experiments and report the following:
  - The hierarchy construction time and the rendering time for each of the accelerators.
  - Among all experiments, which method is faster?
  - Explain why that method is faster?

Task 4: Accelerate Your Own Scene (2 pts)

Use the scene you created in Step 2, choose any one of the settings above. Please report the rendering time and tree construction time for your scene.

Step 5: Submission (1 pt)

What to submit, a zip file contains:

- For task 1:
    - Any modified files for task1, including materials/compmat.cpp, materials/compmat.h, api.cpp, etc.
- For step 3:
    - Five modified coffee.pbrt files: 3_kd.pbrt, 3_bvh_sah.pbrt, 3_bvh_hlbvh.pbrt, 3_bvh_mid.pbrt and 3_bvh_equal.pbrt.
- For step 4:
    - Your accelerated scene file 4.pbrt.
- A report:
    - briefly explain how you implement Task 1.
    - Answers to Task 3.
    - Answers to Task 4.
    - A Google Drive link for downloading the mesh files used in Task 2:
        - The mesh file can be .ply file(s) or .pbrt file(s) converted from .obj using obj2pbrt tool provide by PBRT.

Late penalty: you will lose 10% of your grade each day you are late, and receive 0 after 5 days.