Inverse Rendering Assignment

Overview

You are provided with a series of 2D images of a wooden plane captured by four known cameras and illuminated by eight known point light sources. The cameras and lights are arranged on a ring above the plane, each at an elevation angle of 60 degrees. The goal of this assignment is to retrieve the diffuse reflectance (or albedo) texture of the wooden plane using the given data and known material properties.

1. Setup and Calibration

You are provided with a `calibration.json` file that describes the setup. The calibration data specifies the cameras and lights as follows:

1.1 Cameras

- The cameras are modeled as simple pinhole cameras with ideal optics.
- **Intrinsic Parameters**: Resolution (`width`, `height`) and focal length in pixels (`focal_px`).
- **Extrinsic Parameters**: Roto-translation using a rotation matrix (`rig_extrinsics_R`, row-wise) and translation vector (`rig_extrinsics_t`), transforming from the rig's reference frame to the camera's sensor frame.

1.2 Lights

- Point light sources are modeled physically. Their extrinsics are defined similarly to the cameras.
- Two light models are supported:
    - **Ideal Point Source**: Defined by the "power" in watts.
    - **Emissive Sphere**: Specified by a "diameter" in meters and "radiance" in $W \cdot sr^{-1} \cdot m^{-2}$.

2. Scene Model

- **Geometry**: The plane geometry is provided in `mesh.obj`. It is a 0.5m x 0.5m plane with a surface normal pointing in the positive y-direction.
- **Material Properties**:
    - The plane's surface reflectance follows the PBR model with known roughness and metallic parameters:
        - **Metallic**: 0.0 (purely dielectric)
        - **Specular Color**: 0.04 (monochrome)
        - **Roughness**: 0.3 (spatially constant)

3. Data Provided

The image data is structured as follows:

3.1 Image Data

- Images are captured from each camera with one light turned on at a time, totaling 32 images (4 cameras x 8 lights).
- Available in:
  - **HDR Format**: `.exr` floating-point images for accurate representation of linear brightness.
  - **LDR Format**: `.png` 8-bit sRGB images tonemapped with gamma 2.2 and limited dynamic range.

3.2 Metadata

- **Depth**: Provided in meters.
- **Normals**: Encoded in world coordinates.
- **Positions**: World positions in meters.
- **Mask**: A binary mask.
- **Note for LDR counterparts**: Normals and positions are remapped to the range of [0,1] via the Formula x1 = (x + 1.0) / 2.0

4. Task

Retrieve the diffuse reflectance (albedo/base color) texture of the plane using the provided data. The output can be in:

- **2D camera space**: Please provide each camera view.
- **UV space**: As defined by the `mesh.obj`.

5. Deliverables

- Short explanation of your approach, key decisions, and any assumptions.
- Python or C++ code that implements your solution. Focus on clarity and readability!
- Albedo texture output in the chosen format.

6. Suggested Approaches

You are encouraged to choose an approach and document your method and rationale. Here are some high-level strategies:

6.1 Differentiable Rendering

- **Use Frameworks**: Mitsuba2/3, Pytorch3D, Redner, etc., for end-to-end optimization.
- **Challenges**:
  - Scene setup and calibration data parsing.
  - Potential need to approximate PBR models if it is not supported natively.

6.2 Photometric Stereo

- **Method**: Estimate the albedo in 2D camera space.
- **Challenges**:
  - Handling specular highlights.
  - Aligning camera views and dealing with non-far field illumination.

6.3 Other Solutions

- **Neural Techniques**: Consider neural rendering or other machine learning-based approaches such as using a pre-trained single shot estimator network.
- **Documentation**: Clearly explain and justify your methodology and discuss the limitations.

Notes

- Creative solutions are welcome, as long as they are well-documented.
- Focus on understanding and accurately parsing the provided data.