Introduction

This document provides a brief overview of WCTE photogrammetry simulations performed in the summer of 2022. The idea of this project is to build on the results of the existing photogrammetry simulation work performed by Nick Prouse. This work involved using the OpenCV's 3D reconstruction library in a Jupyter notebook environment to create simulated images of photogrammetry LEDs from the perspective of Sony A7R-IV cameras positioned inside the detector. The main purpose of these simulations was to determine an ideal camera configuration, ensuring each LED was visible from at least two cameras. The main candidates were 6-camera (4 barrel, 2 cap) and 8-camera (all corners) configurations, of which the 8-camera configuration was selected.

The mPMT photogrammetry LEDs for this simulations were treated as points – 12 points in a ring surrounding each mPMT assembly. A sample of the results is shown in Figure 1.

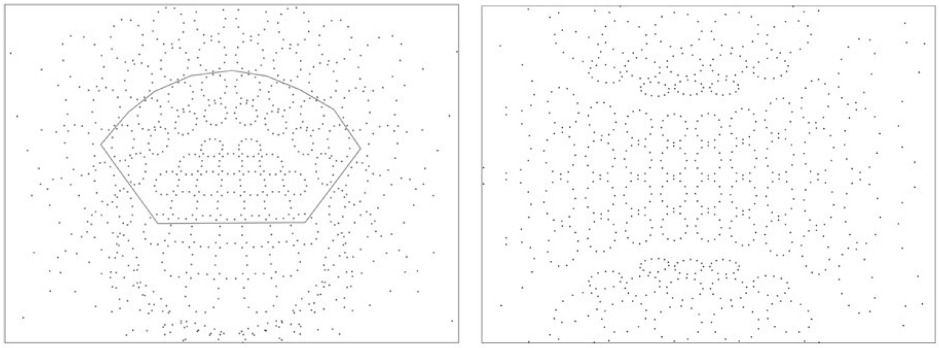

Figure 1: Simulated images of WCTE photogrammetry LEDs from Nick Prouse's photogrammetry simulation. Left: camera in a corner position (usable area shown). Right: camera in a barrel position.

I proposed to use Blender, a 3D computer graphics software tool, to create a more realistic rendering of the photogrammetry LEDs. The key motivation was to account for the angular distribution of light from photogrammetry LEDs for a more accurate image simulation, by treating each LED as a light cone, instead of a point source (see Figure 2). This would result in fewer LEDs being captured by the camera.

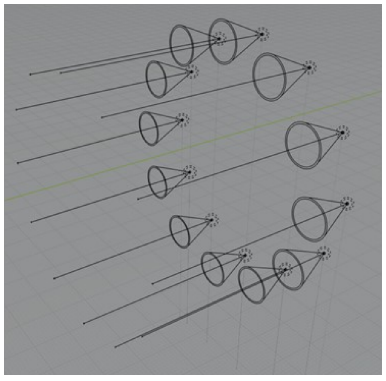

Figure 2: Blender layout view – light cones around 12 mPMT photogrammetry LEDs

Nick Prouse's latest OpenCV Jupyter photogrammetry simulation can be found on his Photogrammetry-Analysis repository. All relevant files from my Blender photogrammetry simulation can be obtained from the WCTE Blender Photogrammetry Sim repository on GitHub.

Setup

The latest version of Blender can be downloaded from here. It can be used to open the main file in the repository, blender_simulations_main.blend. My suggestion is to use the Neuprime desktop to run Blender on, rather than a personal computer, allowing the renders to run overnight and taking advantage of the Neuprime's graphics card.

Summary of the rendering pipeline

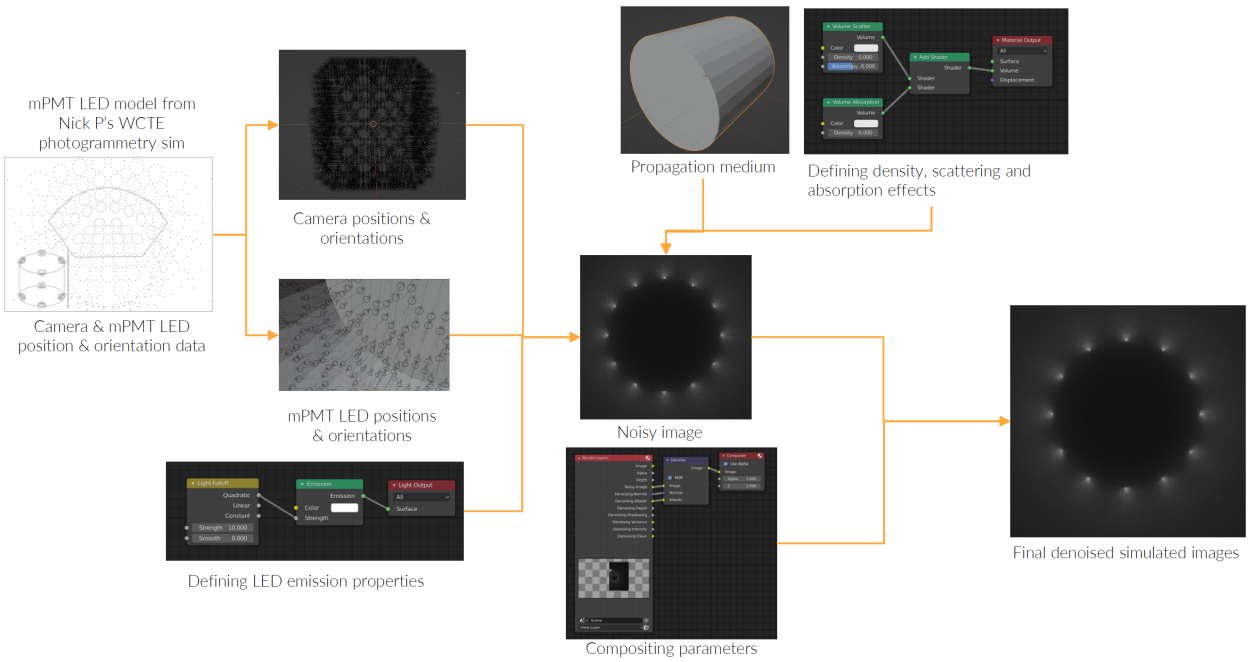

Figure 3: Diagram illustrating the entire process for creating renderings of the mPMT photogrammetry LEDs in the WCTE detector

Refer to Figure 3 showing a flowchart of the Blender rendering process.

All camera and LED positions and orientations were extracted from the original Jupyter notebook, stored in led-positions-rotations.txt and camera-positions-rotations.txt respectively. They were converted and imported into the main blender file through the use of an importing script I wrote, located in the Scripting tab in Blender.

In our application, the rendering process can be treated as a combination of five sets of parameters – these are parameters for the LEDs, the propagation medium, the camera, the rendering, and the post-processing. As a simple illustration of this process, let's look at one of the first renders I made. Refer to Figure 4 below.

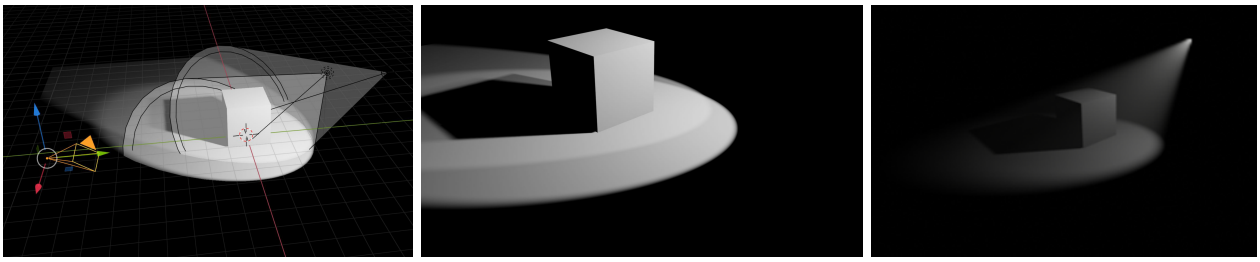

Figure 4: Simple demonstration of rendering a cube with a light source in Blender. Left: View of the setup. Middle: Rendering in Eevee without volumetric lighting effects. Right: Rendering in Eevee with a volume for light propagation.

The idea is to first create a scene – in this case a cube and spotlight sources were created, and a camera was added facing the cube. The rendering engine was selected (more details on this will follow), and the rendering shown in the middle image of Figure 4 was produced. Notice here that the light source itself is not visible, only its effects on the ground and the cube show up in the camera. There are no particles in the air to allow for scattering of the light. In order to see the cone of the spotlight itself, a volume must be defined with parameters describing how light will propagate through it (more on this will follow too). The end result

is the rendering on the right in Figure 4 – a light cone in the air is clearly visible. This is the effect I was trying to create in the renderings of WCTE photogrammetry LEDs.

Having presented the five components of the rendering process and given an example, I will now go into more detail on each one.

The following tabs are relevant to this project:

- Upon opening the main Blender file, you will be presented with the **Layout** tab. This is where all of the components of the simulation can be viewed in 3D.
- The **Shading** tab is used to access nodes for conrtolling properties of the LEDs and the propagation volume.
- The **Rendering** tab can be used to render an "animation" with multiple cameras, which for our application allows for rendering multiple cameras sequentially without any adjustment.
- Post-rendering effects can be applied in the **Compositing** tab. For this project, I have only tried applying a denoising filter, which required collecting denoising data during the render.
- Finally, my importing script is located in the **Scripting** tab.

Rendering Engine

Blender has three rendering engines:

- **Workbench** is the default engine used during rapid development, which uses very limited lighting effects.
- **Eevee** is used for realtime rendering (for videogames, animations, etc.)
- This leaves us with **Cycles**, which is used fo rcomprehensive realistic rendering. I switched to using this one early on in the project, to produce the most realistic light scattering and absorption effects.

Rendering parameters

Selecting the Rendering Properties  icon in the toolbox will give access to numerous options for controlling the rendering process. I will summarize the ones I have modified below:

- **Samples**. This controls the number of paths that are traced to produce the final render. The **Viewport**'s samples refers to the number of samples taken for rendering in the viewport (the initial result you see in the **Shading** tab), and the **Render**'s samples control the final rendered image. The more here the better, I used 3000 for final images and kept Neuprime running overnight.
- **Denoise**. Introduced in a newer Blender update, this option applies a denoising filter after rendering. I set this option to off for the final images – while it did reduce the grainy effect from not having enough samples, it blurred the image in a non-uniform / unpredictable way.
- **Light Paths – Max Bounces**. The number of times light reflects off of surfaces. Since there are no objects in the WCTE simulation to reflect light, this option has no effects on the simulation, apart from slowing it down. I kept this at 1 for the final images.

Camera parameters

The camera parameters, located under the Object Data Properties  (make sure a camera is selected in the **Layout** tab), are a lot more extensive than what I had time to explore. Additional libraries can be added to control more distortion parameters, including potentially a fisheye model – this could be a future avenue to explore with the project. I modified only the **Focal Length**, setting it to 14 mm to match the Samyang lens. The sensor size was kept at 36 mm to match a full-frame DSLR camera. Note that changing the options for one of the cameras will effect the other cameras as well, because I grouped them together.

LED parameters

For the LED parameters, I took advantage of Blender's node system for controlling the properties of objects. Nodes are a way of applying the different shaders, effects, and other object properties that Blender offers. Different nodes can be combined and routed together to create the desired effect. To access these, visit the **Shading** tab and select an LED. You will see a viewport output window on the top half, and the nodes windows on the bottom half, where you will be presented with a set of nodes with the effects added to that LED. Changing the properties of a single LED changes the properties for all LEDs in the simulation, because of the way I have grouped the LED objects together (same for the cameras).

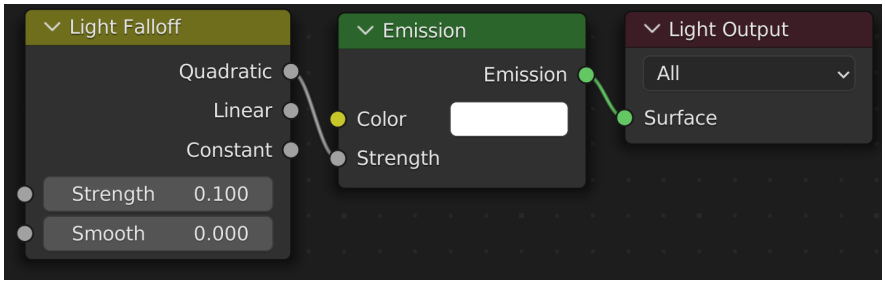

Figure 5: Nodes for configuring properties of WCTE mPMT photogrammetry LEDs

As you can see in Figure 5, the Light Output node is connected through an Emission node to a Light Falloff node set to the quadratic setting. This ensures that the Light Falloff replicated the standard inverse square law. This quadratic falloff node is one of the defaults offered by Blender – it is possible to create custom nodes and define formulae for the light's path to follow. This can be used to account for other effects, or to model a non-uniform dispersion / LED beam profile. The Emission node can be used to control the color of the LEDs. I found that setting this to white and the absorption in the volume to blue produced a good result, though eventually a green LED wavelength should be used.

Additionally, a few more LED settings are available within the Object Data Properties dropdown. The following are relevant and have been experimented with:

- Radius controls the side of the spotlight. Set to 0.0025 m in the final images.
- Max Bounces controls the maximum number of bounces that will be visible in the render. Because there are no visible objects in the scene, this parameter did not seem to have a significant effect on the rendering. It is set to 1024 in the final images, though it could have been 1 with the same effect.
- Spot Size determines the angle of the light cone emanating from each LED. It was set to 45° in the final images to roughly match the angle from light pipes obtained by experimentation.
- Blend controls the softness of the edge of the LED light cone. A value of 0.150 was used in the final images.

Note that the Power and Color options you see in this menu have no effect on the LEDs, since these are controlled via the system of nodes shown in Figure 5.

Volume Parameters

The parameters of the propagation volume are controlled in the same way the LEDs' is – through the nodes system in the **Shading** tab. After selecting the Cylinder at the bottom of the objects dropdown, the nodes shown in Figure 6 will appear.

We can see that I have chosen a combination of two effects to be added to the Material Output. These are the Volume Scatter and Volume Absorption nodes. The Volume Scatter node will add scattering effects, and has two controllable properties: The density corresponds to the strength of this effect, and was set to 0.001 in the final images. An non-zero anisotropy setting can make light directional – a negative value will favor light rays in the backward direction opposite the camera, and a positive value will favor light rays in the forward direction towards the camera. Thus having a positive value causes LEDs which are pointing towards

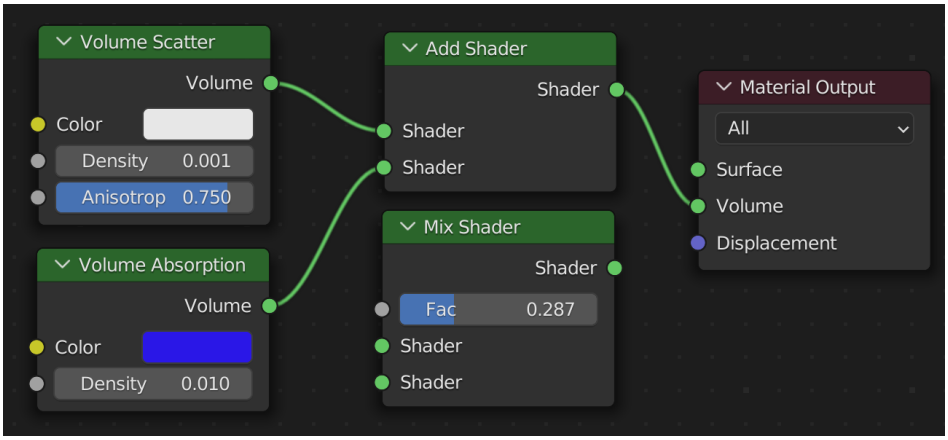

Figure 6: Nodes for configuring properties of the volume for light propagation

the cameras to be favored over those at the sides, aimed in a direction more perpendicular to the camera. A value of 0.750 was used for the final images, though based on tests conducted in the mPMT tub at TRIUMF, which I reported on at the August 30 Calibration meeting, a value of 0.750 could be underestimating the anisotropy.

Results

To create a render of a single image using the camera that is currently selected, one can simply go to Render → Render Image. If a group of images from a set of camera configurations needs to be taken instead, the Rendering tab can be used to create an animation, in which each frame is from the perspective of a new camera. On the bottom half of the Rendering tab, an animation bar will show all of the frames. To add a new frame with a new camera, first select the camera using the Outliner dropdown tab. Next, select a frame by clicking on the frame number. Press CNTRL+B to apply the selected camera to the selected frame. The entire animation can now be created using Render → Render Animation.

All of the relevant renders that I made can be found on the GitHub page for this project. This includes two types:

- Renders of the WCTE mPMT photogrammetry LEDs with a camera at the centre of the detector looking at the top of the cylinder and cameras in the corner positions can be found in the August 16 and August 17 folders. The latest and best images are located in the August 17 folder. These were created with the parameter values described in the sections above. Some samples shown in Figure 7.

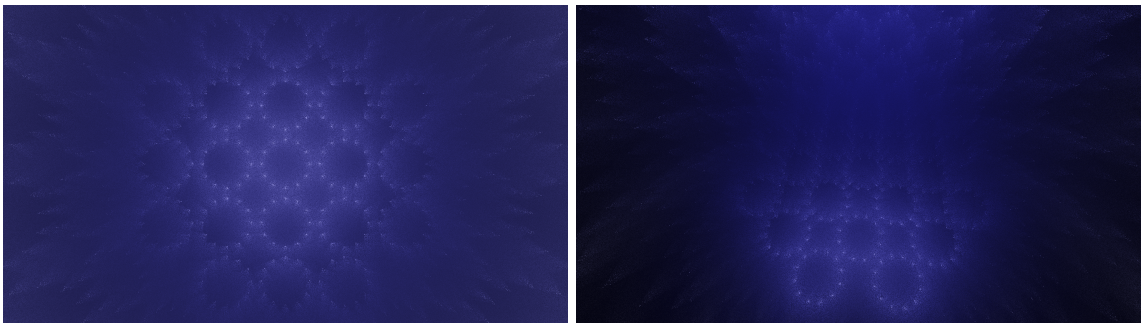

Figure 7: Sample of results from full renderings of the WCTE detector's mPMT photogrammetry LEDs. Left: The top of the detector with the camera positioned at the detector centre. Right: Camera positioned in one of the detector corners.

- Renders of a single LED located 1 m away from the camera rotated at different angles to the camera can be found in the August 22 folder. The presentation comparing these with images taking using a submerged light pipe in the mPMT tub at TRIUMF can be found on the August 30 Calibration meeting page.