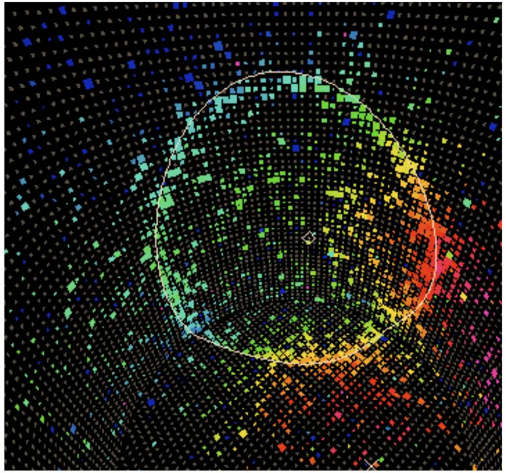
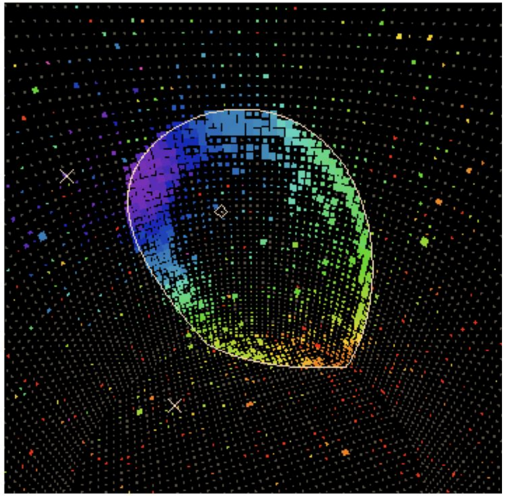
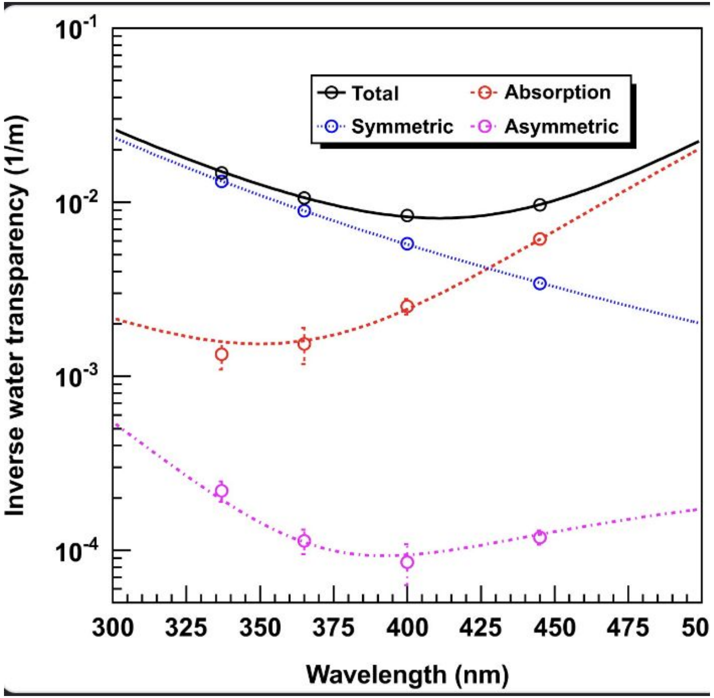
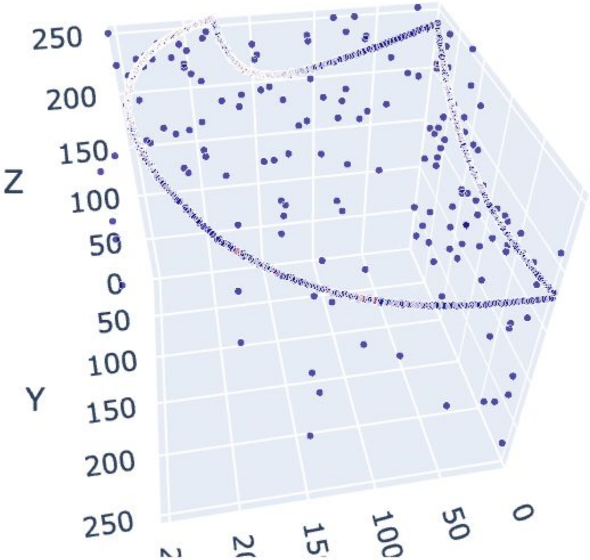
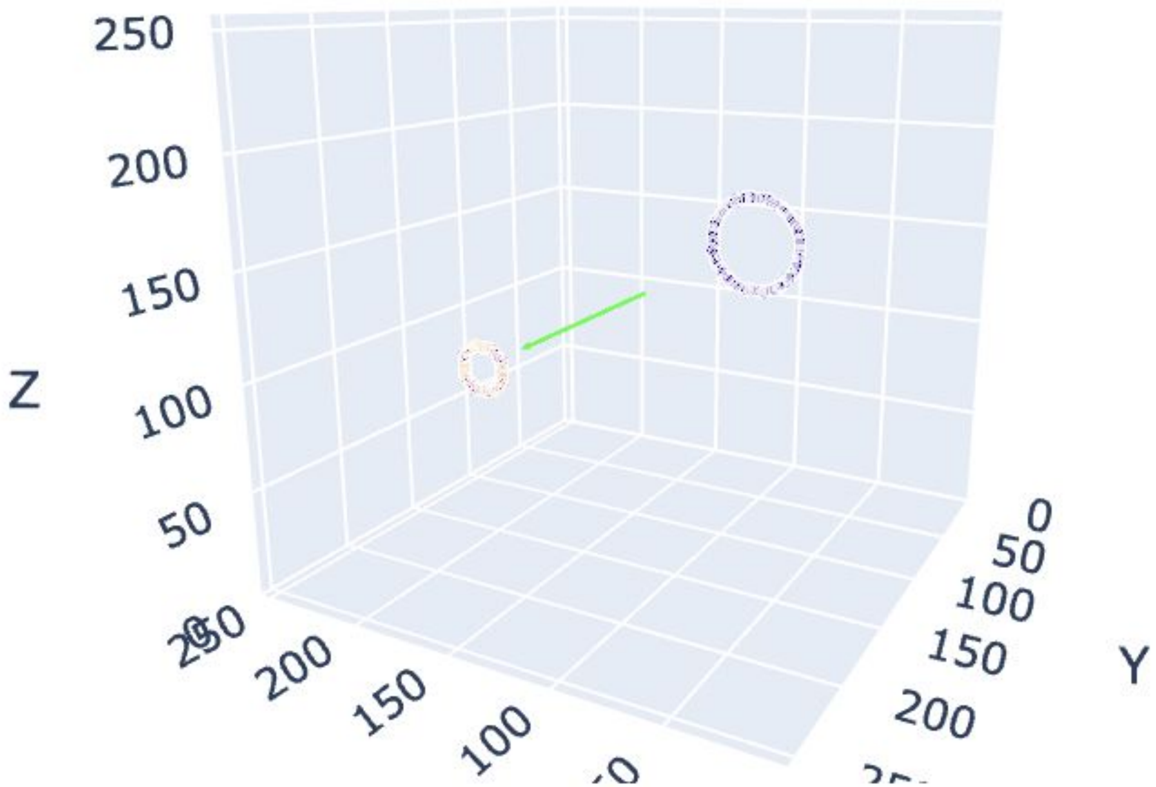




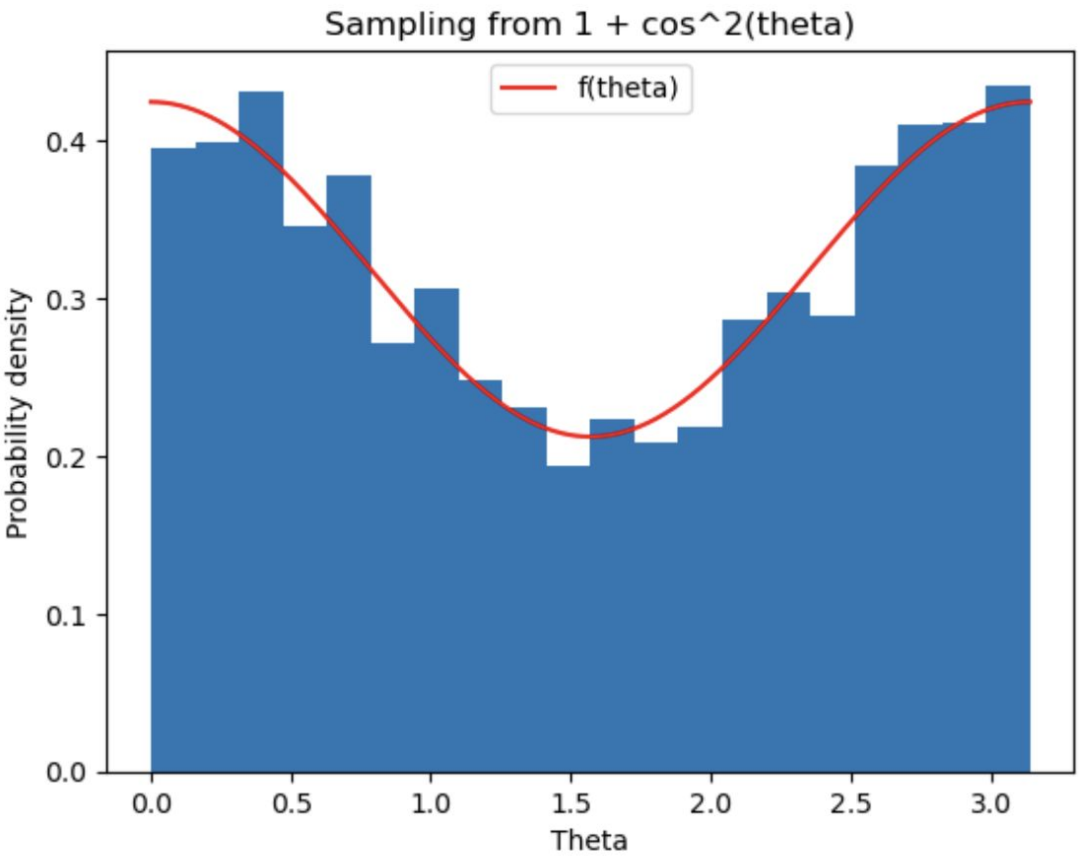
$$I = I_0 \frac{8\pi^4 N\alpha^2}{\lambda^4 R^2}(1 + \cos^2\theta)$$





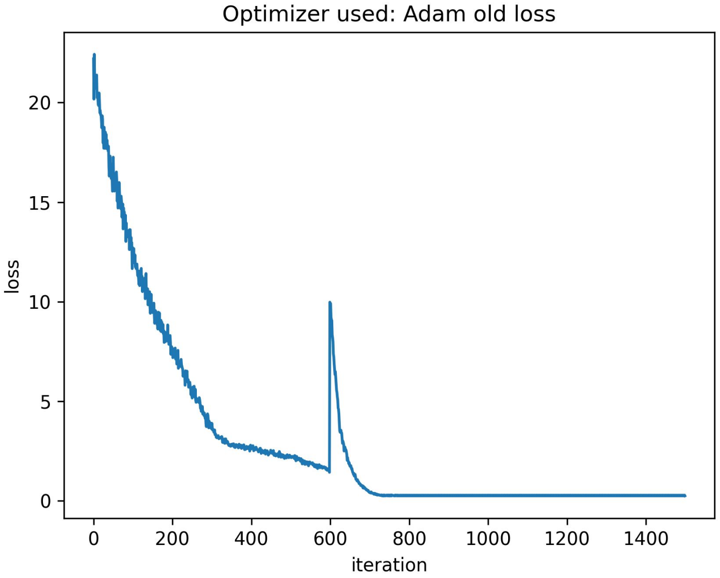


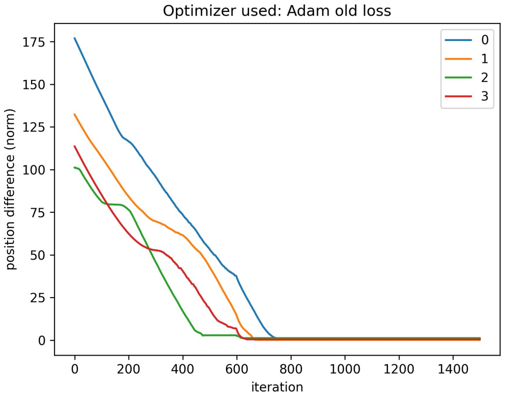
Implemented reflection in differentiable forward

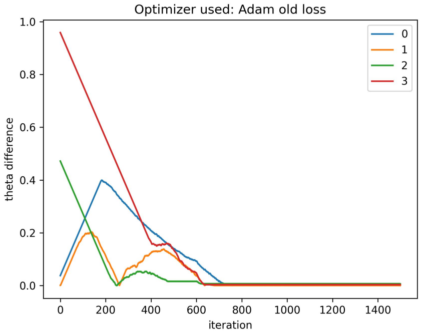


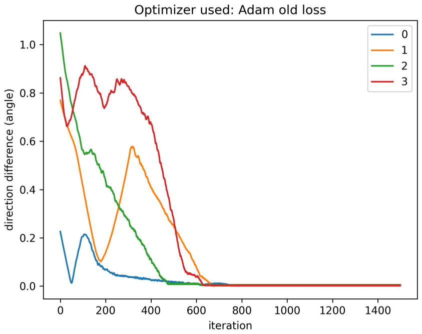
Check 1: Theta for scattering distributed as one would expect

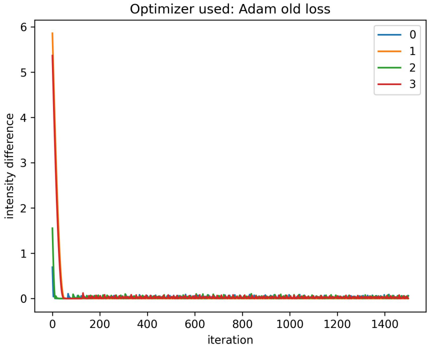
Adam training with old loss



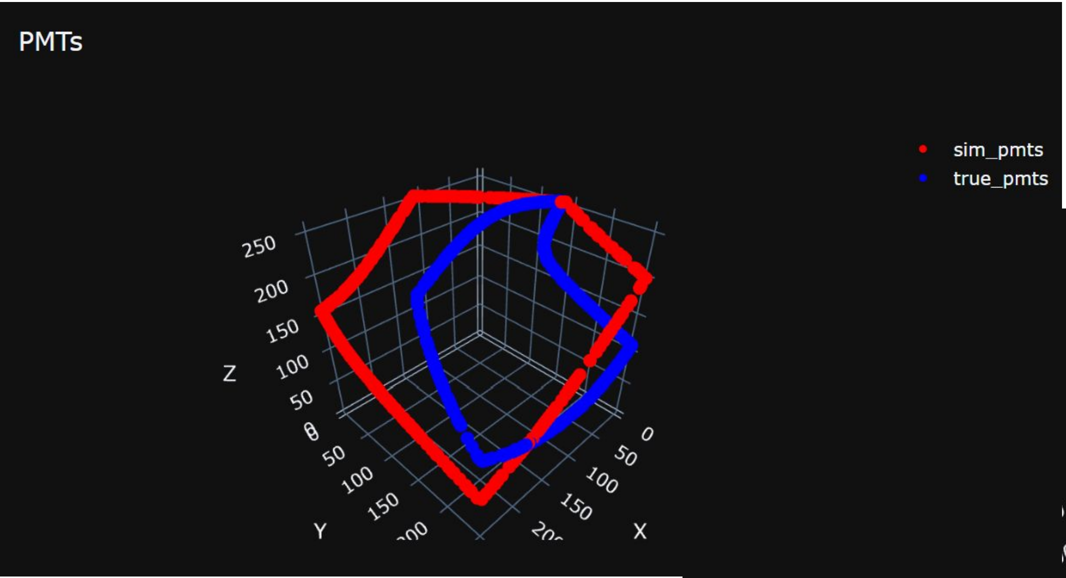


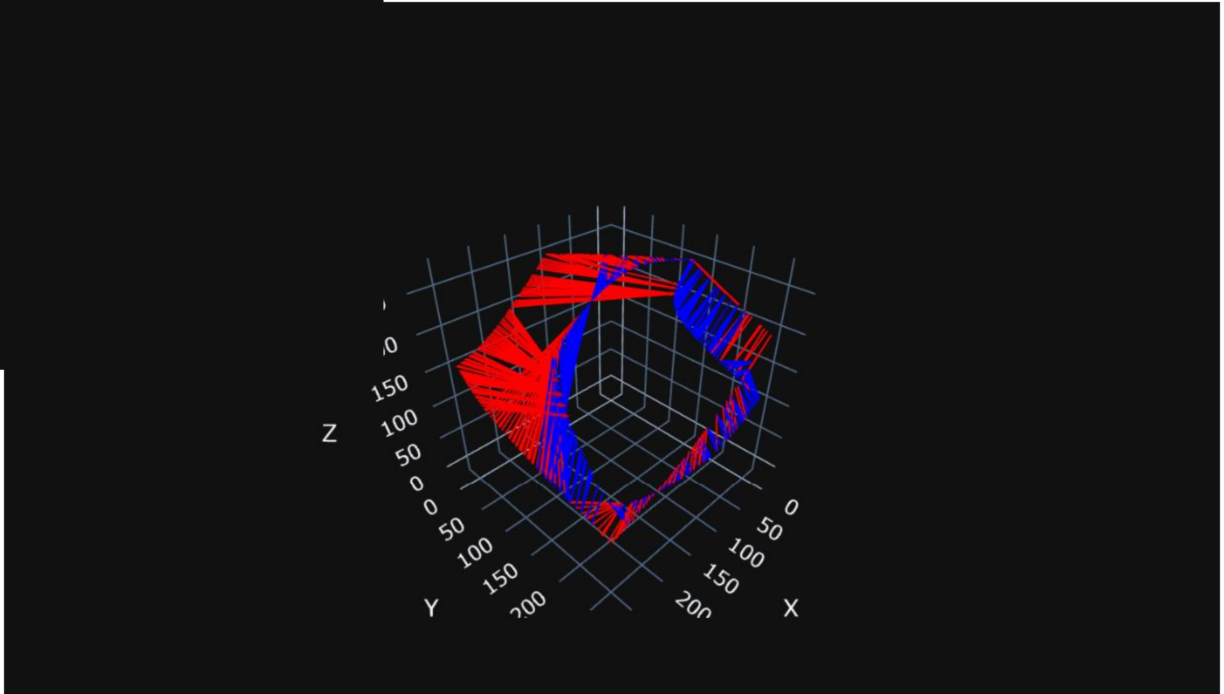






Arrows indicating closest pmts from true to hit





Explicit/Analytic DDSim

Project Goal: Demonstrate that Analytic DDSims are well suited for WC reconstruction & more in general, to discuss that there is no fundamental impediment to develop DDSim particle physics frameworks aimed at replacing GEANT4 (but offering diff capabilities).

How to: Create minimal detector toy model and use it to discuss & demonstrate how this would work. We choose ‘taichi’ as the language for this implementation as it does many of the things we want to deal with, e.g. propagation through space voxelization, autodiff, sparse computations (but it has obvious shortcomings as this is really not a framework meant for particle physics).

Previous workshop: toy-detector geometry was implemented, photon propagation non-diff forward was also implemented (together with some visualizations).

After the workshop we continued working in two directions:

- Cesar: Modified non-diff forward to diff-forward, changed the loss to achieve convergence and deal with hyper parameters. Demonstrated that simultaneous reco & calibration is possible. Code implementation meant to be fast and OK-ish.
- Omar: Focused on refactoring the code, making it more scalable, also implementing better solutions for same issues I face (e.g. using standard ML optimizers rather than doing optimization by hand).

In the workshop: Facing next-steps (Cesar)

- First days we focussed on addressing how to continue, so far most of the implementation was ~easy, because we could ignore the issue of randomness, but we needed to deal with it.
  Decided to go for pre-setting random coins, use them to run the forward in differentiable way.
- Afterwards, implemented non-diff fwd for reflections & scattering first. Looking into evt displays & other distributions things looked OK.
- Changed code to be “differentiable” on the surface it all looked OK, achieving convergence.
- However further tests revealed convergence was only achieved because non/reflected & non/scattered photons had non-zero grads.
- Focussed on investigating gradients for reflected cases.

In the workshop: Facing next-steps (Cesar)

- Realized taichi implementation for that case is much more difficult than I originally expected. Primary difficulties are:
    - In contrast with previous cases propagation is not now static but dynamic (e.g. we have to take different decisions depending on if something happens or not). This is very bad for differentiability. Requires quite significantly different way of implementing the forward (and it is generally quite terrible to debug).
- Looking to loss 1D profiles to check that both the loss and the gradients look OK.
- Yesterday achieved good gradients for position variables with full reflection, however null-gradients for time variables. Realized the issue was unrelated to reflections, pulled code from Omar, now working to achieve good gradients for time as well.

Next steps (Cesar)

- Get good gradients for full reflections in both position and time variables.
- Get good gradients for non-full reflections (requires checking that if conditions for the coins do not break the gradient).

- Same (but for scattering).
- Move to realistic loss (not looking at true information).

- Define list of plots/metrics we would like to report with the toy-model (e.g. thinking what might be a paper-skeleton).
- Define list of datasets to achieve them (e.g. w & w/o reflection & scattering, etc).
- Make nice plots…

In the workshop (Omar):

- This workshop involved a lot of code refactoring: Things that were changed include:
    - Adam optimizer fixed and running
    - Bug in detector geometry fixed
    - Taichi Fields definitions refactored to fix gradients problem (+use taichi sparse fields for efficiency)
    - Old loss was altered to get directional information even without any photon direction information given by fixing the time definition
    - New loss:
        - Implemented finding closest hits to other hits in taichi (which we could explicitly see in the visualization)
    - We generate simulation + “true” data. Removed all the true and allowed generating once and saving/loading from a file
    - Successfully created a new loss with no single photon information (compares hits to hits)

New Loss:

- First get closest true hits to sim and closest sim hits to true (We use this for positional loss):
    - For each photon in sim, we take the loss of its final position with the closest true pmt
    - For each true pmt, we take the loss if it's position with all the positions of the photons in the closest sim pmt
- We then bin the time of arrival of each pmt by some delta_T. For each bin, we sum up the intensities of each photon in that bin and average the time of arrival.
- We then have two losses for hittimes:
    - First, per pmt, we take the sum of all intensities and compare true pmt vs sim pmt (do the same for time as well)
    - Second, per pmt, we compare every true bin with the exact same sim bin
- This loss is finally working now (we get gradients!)

Next steps (Omar)

- Test the new loss and make sure it works as expected
- Try some hyperparameter combinations and altercations of the loss to get better and more robust convergence
- Work with Cesar in trying to do full stochasticity for reflections and scattering
- Add a layer of doing particles instead of rings (we define a particle’s initial values and generate rings from that while making sure it’s differentiable)