Dark Matter Physics Simulation & Analysis for the Dark Photon Milestone 5

By: Nikhil Chaba, Samuel Rock, and Jacob Woods

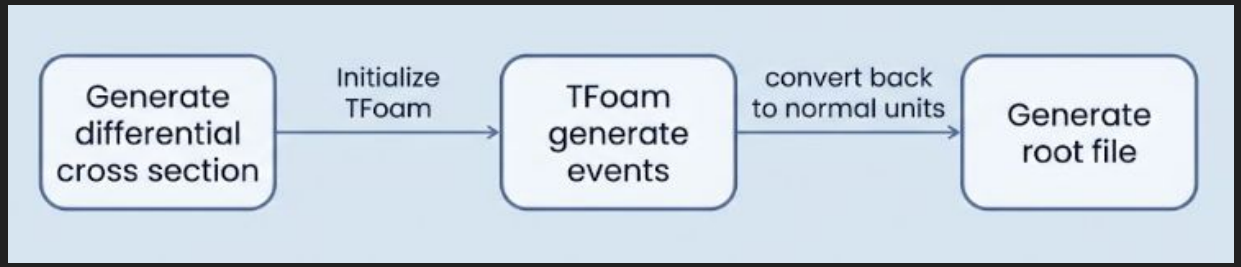
Pipeline Overview

We are combining different research papers and notebooks to produce this new pipeline:

[1] H. D. Davoudiasl, & R. M. Marcarelli, & E. T. N. Neil. “Displaced Signals of Hidden Vectors at the Electron-Ion Collider,” *Phys. Rev. D*, vol. 108, no. 7, Art. no. 075017, Oct. 2023. https://arxiv.org/abs/2307.00102

[2] “Constraining Long-Lived Gauge Bosons at the EIC and a Muon Beam Dump,” Python notebook

[3] R. Mighty and A. R. Kanuganti, "Machine Learning for Invisible Dark Boson Searches at the Electron-Ion Collider," Sep 2026. https://arxiv.org/abs/2609.22484v1



Generating differential cross section

The MAIN equation [2]

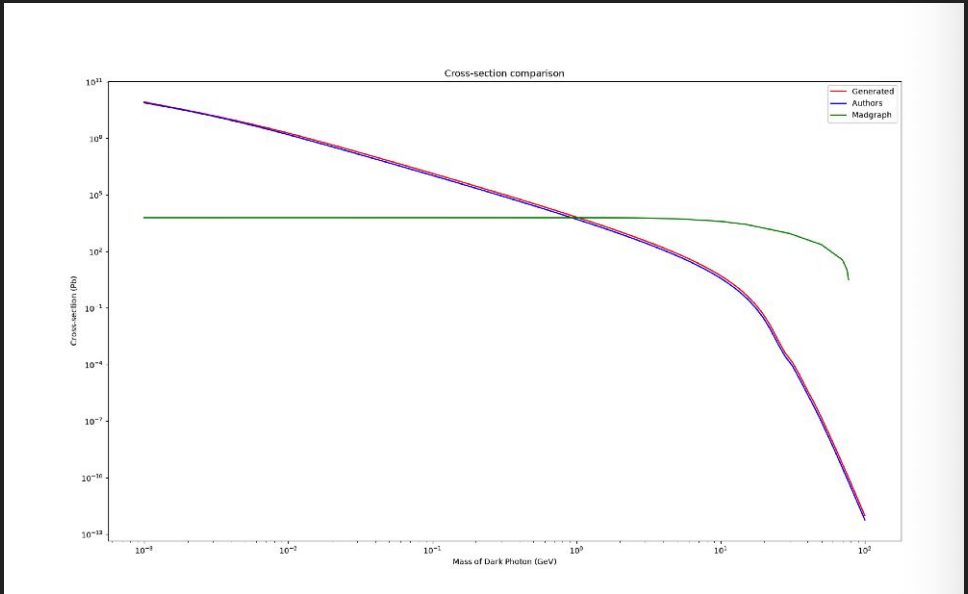
$$d\sigma/(dE_k\, d\theta_k) = (g^2\alpha^2/4\pi) \cdot (|k| / (|p| \cdot V)) \cdot (\chi / 2t_{min}) \cdot \mathcal{A}$$

Two step calculation

For a given vector energy and theta angle

Bugs/Fixes

- NaNs
- Cross section values produced were too small to make sense



Summary on generate_vector.cpp

- We examined [3], which investigates whether a boosted decision tree can improve signal selection over regular cuts, and whether adding the Mandelstam variable t (as $t = -(P_i - P_f)^2$, where P_i and P_f are the nucleus's initial and final four-momenta) helps.

- From the paper's research, we looked at generate_vector.cpp, because it contains the simulation, calculates cross-section, and generates events.

[3] Mighty and Kanuganti, *Machine Learning for Invisible Dark Boson Searches at the Electron-Ion Collider* (2026). https://arxiv.org/abs/2609.22484v1

Summary on generate_vector.cpp

The main idea of the program was:

- Simulates: $e^- + Au \rightarrow e^- + Au + V$, where V is the invisible dark vector boson.
- Uses: 18 GeV electron beam and a 100 GeV-per-nucleon gold beam, and TFoam
- Studies: dark-boson masses from 0.01 to 10 GeV, with a benchmark electron coupling of 10^{-4}.
- Produces: total cross-section estimates, electron kinematic distributions, and weighted event record and histograms in ROOT format.
- We focused on their use of TFoam, the main class of the multi-dimensional event generator FOAM.

General explanation of TFoam:

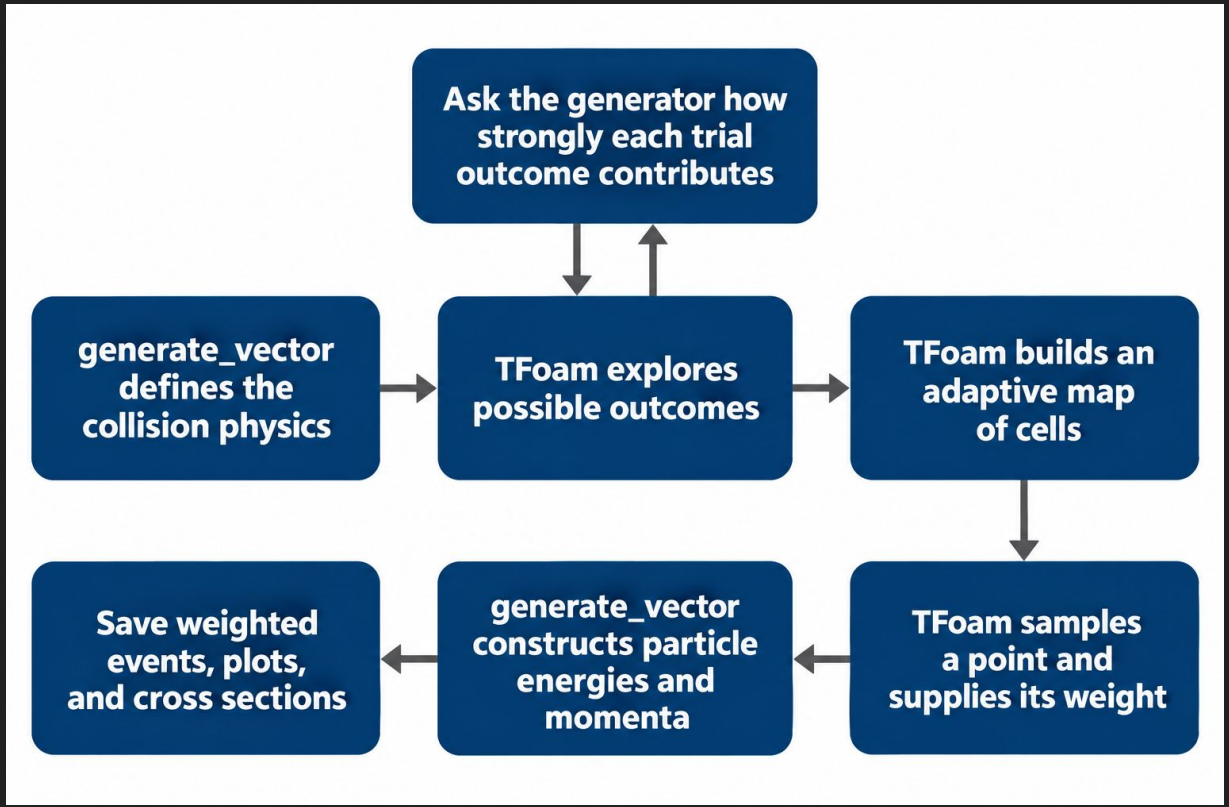
- TFoam is a tool that helps a simulation choose which possible particle collisions to generate.
- It uses random sampling, called Monte Carlo sampling.
- Adapts that sampling to a distribution supplied by the user.
- It can also estimate the integral of that distribution

  The interaction’s total cross section, a measure of how likely the process is to occur.

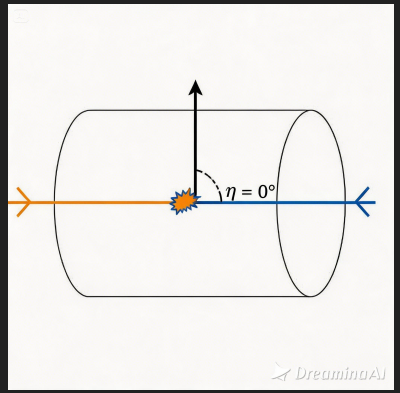
How TFoam is used in generate_vector.cpp

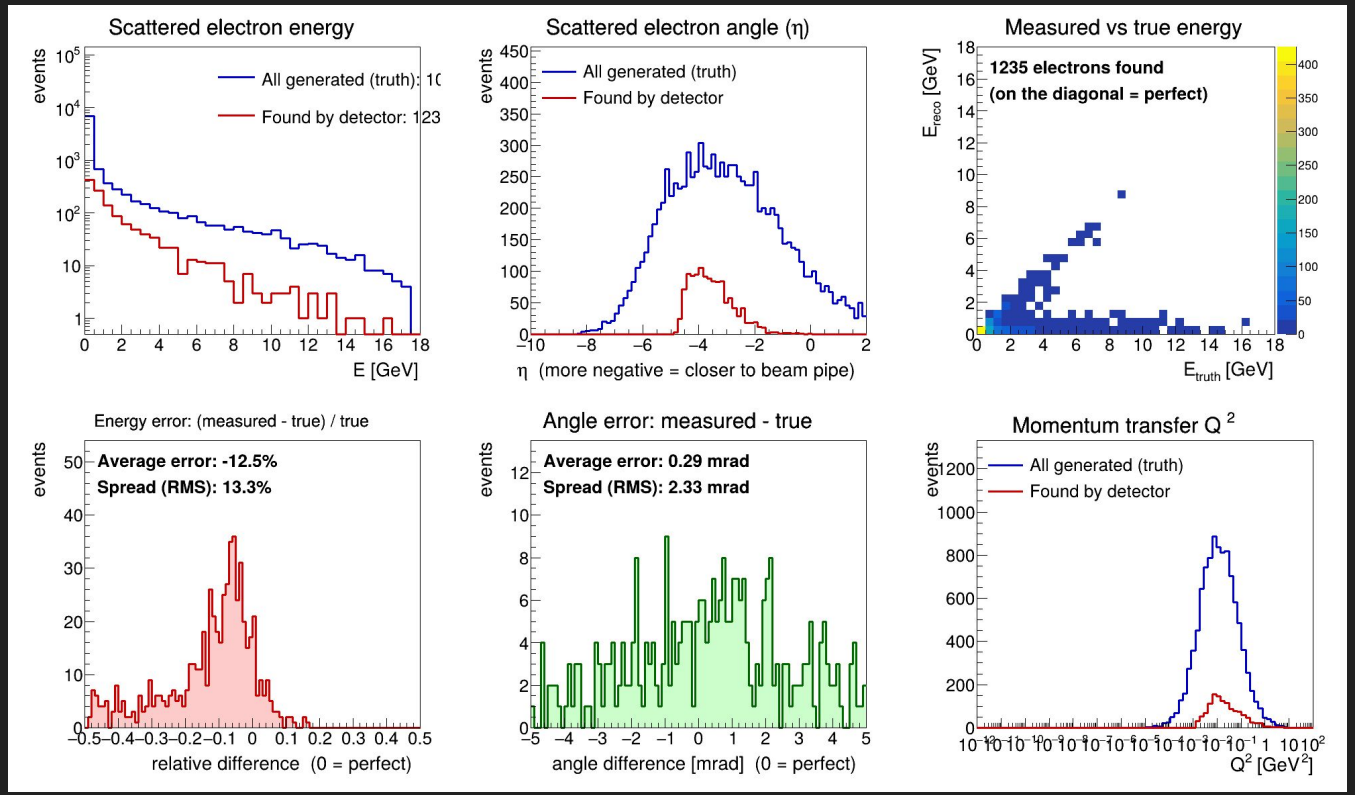
- Imagine a map of all the possible collision outcomes. Some regions contribute strongly to the interaction, while others contribute very little.
- Sampling the entire map uniformly would spend considerable effort in regions with little contribution.
- TFoam first explores the map, dividing it into rectangular regions called cells. It refines the division around important features, such as sharp peaks, then uses that map to guide later sampling. The collection of cells gives the method its “foam” name.

TFoam process diagram:



Results


η = 0°
DreamingAI


Scattered electron energy
events
All generated (truth): 1(
Found by detector: 123
E [GeV]
Scattered electron angle (η)
events
All generated (truth)
Found by detector
η (more negative = closer to beam pipe)
Measured vs true energy
1235 electrons found
(on the diagonal = perfect)
E_reco [GeV]
E_truth [GeV]
Energy error: (measured - true) / true
events
Average error: -12.5%
Spread (RMS): 13.3%
relative difference (0 = perfect)
Angle error: measured - true
events
Average error: 0.29 mrad
Spread (RMS): 2.33 mrad
angle difference [mrad] (0 = perfect)
Momentum transfer Q²
events
All generated (truth)
Found by detector
Q² [GeV²]

Milestone 5 Task Matrix

Task	Completion:	Nikhil	Sam	Jacob	To do:
Integrated a Machine Learning research paper to develop alongside the Weizsacker-Williams approximation	90%	-	1/2	1/2	We have the cross section being calculated, we want to use the machine learning paper to generate events using the cross section
Tested the Weizsacker-Williams approximation codebase	100%	1/3	1/3	1/3	-
Analyzed a gold e- collision at the reconstruction/root level produced from Machine learning generator and approximation codebase	80%	1/2	-	1/2	We have ran successfully creating a root file, we now want to test different results to confirm the reconstructed root file is correct
Wrote user manual	10%	1/2	1/2	-	Continue updating, add a knowledge base section

Milestone 6

- Task 1: Finish the updated pipeline
- Task 2: Finish analyzing a gold -e collision using new pipeline production
- Task 3: Finish writing user manual with given format for research paper

Milestone 6 Task Matrix

Task	Nikhil	Sam	Jacob
Finish the updated pipeline	1/3	1/3	1/3
Finish analyzing a gold -e collision using updated pipeline	1/3	1/3	1/3
Finish writing user manual	1/3	1/3	1/3

Thank You