DOM Efficiency with Muons Overview

Created by Alexander Detkov

Introduction

Python Scripts

The analysis is split into two python scripts and one python module. The module, called *icepy.py*, consists of useful functions and classes that aid in the filtering of tracks and their analysis. The two python scripts, *TracksGenerator.py* and *DOMEfficiency.py*, split the investigation into two processes: filtering and analysis respectively. The purpose of this will become clear in the following sections.

Questions and Concerns

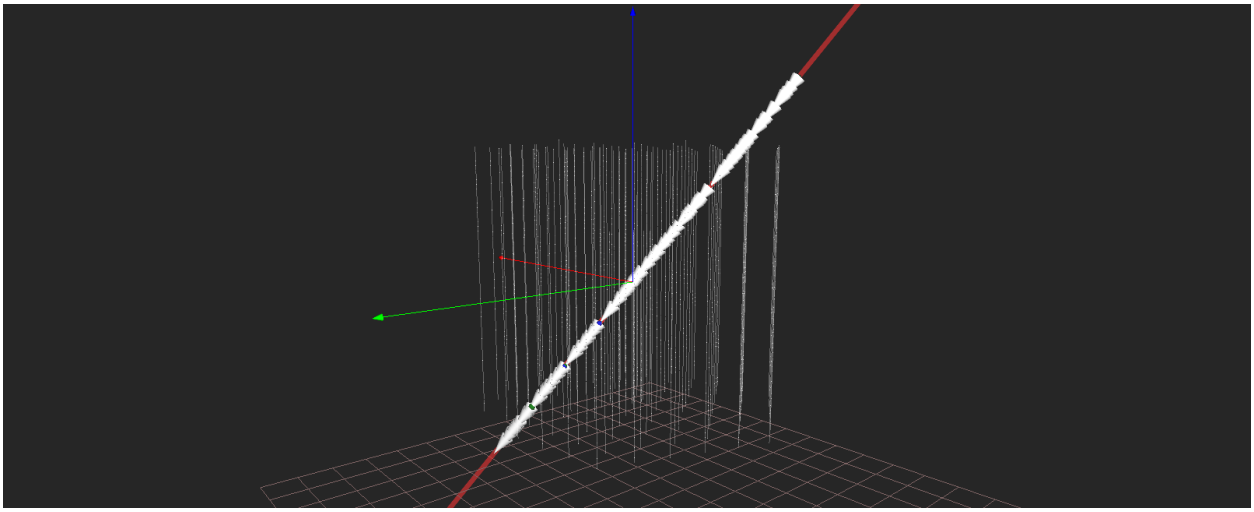
Feel free to email me anytime at detkov@ualberta.ca

TrackGenerator.py

Overview

The purpose of this file is to generate custom “Track” objects defined under *icepy.py*. This script takes raw .i3 files and outputs a .pickle file of the filtered “Track” objects. The .pickle file is then analyzed by the *DOMEfficiency.py* script. The purpose of splitting the filtering and analysis scripts is that reading and filtering .i3 files is a time-intensive process and using the .pickle approach allows for efficient analysis. In short, this approach is to facilitate repetitive analysis of the same data set. We now describe the filtering process done by *TrackGenerator.py*.

Frame Filtering

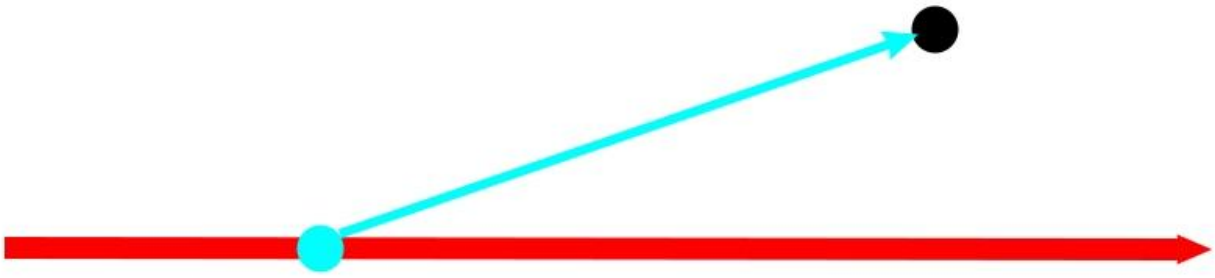


1. Only consider frames where only one muon passes within 100 m of the detector. Only 17% of physics frames remain.

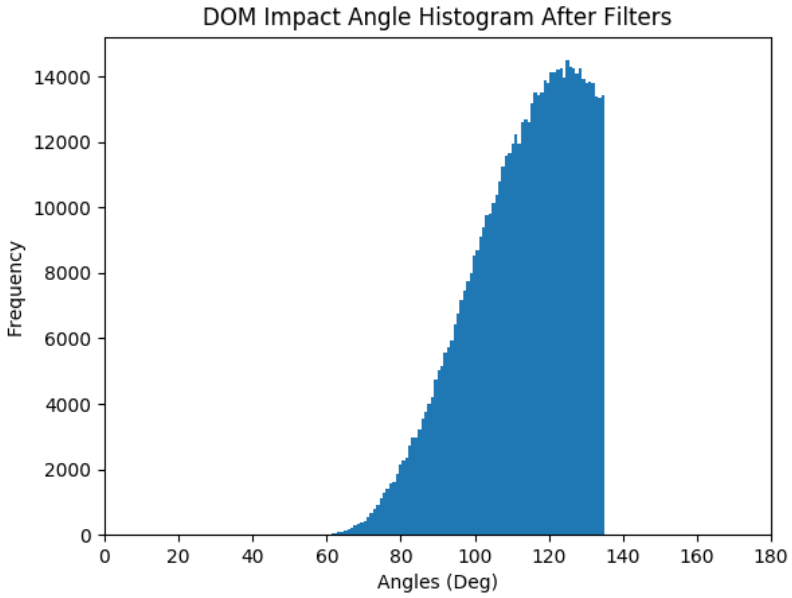
Track Filtering

1. Cutting 50m from the end of the muon track.
    a. Cherenkov emissions and stochastic losses originating from the cut track are ignored.

DOM Filtering and Manipulations



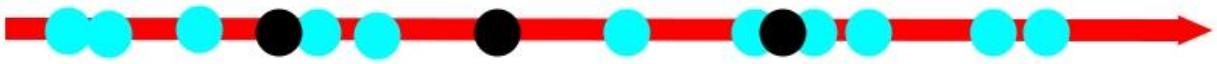
1. Ignore DOMs that cannot be theoretically hit by Cherenkov radiation.
2. Ignore DOMs where the distance the Cherenkov light has to travel is less than 20m or greater than 100m. Only 1.7% of DOMs remain after filtering for Cherenkov distance.
3. Ignore DOMs positioned such that the Cherenkov light emitted by the track passes through the dust layer at z=[-150, -50]. After filtering, 90% of DOMs remain.
4. DOMs that receive Cherenkov light at an impact angle >135° are ignored. After filtering 56% of DOMs remain.
5. DOM pulses manipulations
    a. DOMs with no pulses are considered to have observed a Cherenkov emission of zero intensity. 62% of DOMs have no pulses.
    b. DOMs with more than one pulses have their charges sumed and intensities computed from their sum.
    c. We filter time residuals between theoretical and observed pulse times and keep only those within a time residual in the set [-100, 100].
6. We are also accounting for impact angle through DOM acceptance value.



DOMEfficiency.py

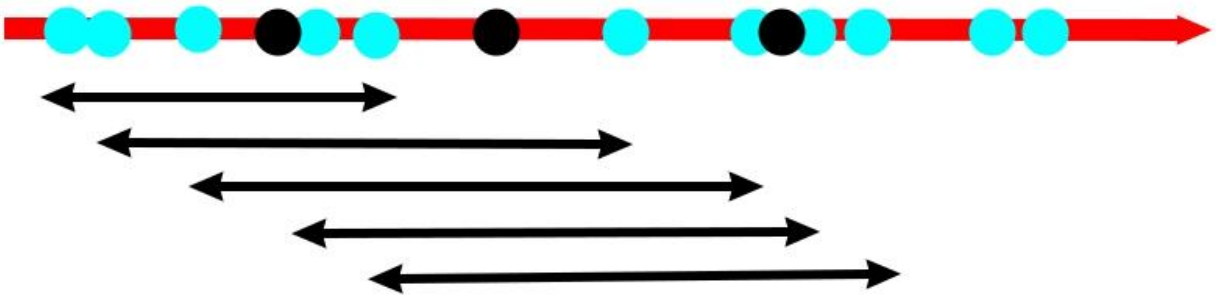
Overview

The purpose of this file is to analyze the .pickle tracks. After *TrackGenerator.py* we have an array of tracks with emissions and losses on the track as shown below. Blue dots are points of Cherenkov emissions and black dots are stochastic losses. We now want to analyze the tracks and draw conclusions. Currently, we are trying to remove stochastic losses from a track by removing bins with high intensity per unit length relative to the mean intensity per unit length of the track. We are currently only removing a single bin. The procedure is outlined below.



Track Analysis and Manipulation

1. First, we calculate the intensity per unit length of the entire track.
2. Then, using a sudo sliding window approach, we create a set of bins of differing lengths such that each bin contains a fixed amount of emissions (5 emissions per bin).



3. We then calculate the intensity per unit length within each bin.
4. Next, we find the bin with the highest intensity per unit length and we determine if it is a high loss bin by comparing its intensity with respect to the track. If it is above a certain threshold we remove it.
5. Histograms and scatter plots are then made comparing the modified tracks to low loss tracks.