DeepCore Veto Methods

Sebastian Euler
RWTH Aachen University

Novel Searches for Dark Matter 2010 Workshop
July 5 2010, Ohio State University
Outline

- IceCube + DeepCore
- Veto: First Ideas
- Stage 1: Velocity Veto
- Stage 2: finiteReco
- Efficiency
IceCube + DeepCore

- **IceCube**: 86 strings with 60 sensor modules (DOMs)
- **instrumented volume**: 1 km$^3$
- **DeepCore**: 6 more densely instrumented strings in the bottom center of IceCube
- **new highQE PMTs**
- **clear ice**

deployment scheduled for 2010/11
Upgoing / Downgoing Tracks

- down-going tracks: atmospheric muons
- up-going tracks: neutrinos
- everything traversing the earth has to be a neutrino

→ usually IceCube analyses select upgoing tracks
Starting Tracks

- downgoing tracks can be recognized as neutrino-induced
- if the interaction vertex can be identified inside the detector

→ look for tracks apparently starting inside the detector
Veto: The Principle

- DeepCore: densely instrumented volume in the center of IceCube
- surrounded by ~4000 IceCube DOMs
- can be used as active veto
- dramatic understatement
- ratio (atm. : ν-ind.) $10^6 : 1$
- veto will not be 100% efficient
Veto: First Ideas

- use ~4000 DOMs around DeepCore as active veto

first ideas:
- reject every event with hits in outer layers

- use “veto weight”:
  - assign weight to each veto hit
  - weight depends on distance to DeepCore
  - add up all weights for event
  - reject events with weight > \( x \)

→ allow events to have few hits in the veto region
Velocity Veto

- more sophisticated method
- implemented as online-filter
- compute center of gravity (COG) of DeepCore hits and vertex time
Velocity Veto

- more sophisticated method
- implemented as online-filter
- compute center of gravity (COG) of DeepCore hits and vertex time
- for each hit in veto region define particle speed:

\[ v := \frac{\text{spatial distance to COG}}{\text{time difference to vertex time}} \]
Velocity Veto

- more sophisticated method
- implemented as online-filter

- compute center of gravity (COG) of DeepCore hits and vertex time
- for each hit in veto region define particle speed:

\[ v := \frac{\text{spatial distance to COG}}{\text{time difference to vertex time}} \]

- hits before vertex time: \( v > 0 \)
- hits after vertex time: \( v < 0 \)
- causally related hits: \( v \approx c \)
Velocity Veto

- more sophisticated method
- implemented as online-filter

- compute center of gravity (COG) of DeepCore hits and vertex time
- for each hit in veto region define particle speed:

  \[ v := \frac{\text{spatial distance to COG}}{\text{time difference to vertex time}} \]

- hits before vertex time: \( v > 0 \)
- hits after vertex time: \( v < 0 \)
- causally related hits: \( v \approx c \)

\[ \rightarrow \text{reject all events with more than one hit with } 0.25 < v < 0.4 \]
more Veto needed…

- implemented as online filter at the pole
- signal passing rate of \(~99\%\)
- background rejection of \(~2\) orders of magnitude
  → need more
- especially low-energy (non-stochastic) muons can sneak into the detector
  (large string spacing, dust layer, ...)
- idea: select tracks with high probability to be neutrino-induced, reject tracks which can also be explained by muon
The finiteReco Algorithm

- find obviously starting tracks among possibly through-going tracks
- input: measured pattern of hit DOMs & reconstructed track (in these studies: true MonteCarlo track)
The finiteReco Algorithm

- find obviously starting tracks among possibly through-going tracks
- input: measured pattern of hit DOMs & reconstructed track (in these studies: true MonteCarlo track)
- look for not hit DOMs in close proximity of the track

likely candidate for a starting track

hit DOM
not hit DOM
The finiteReco Algorithm

- find obviously starting tracks among possibly through-going tracks

- input: measured pattern of hit DOMs & reconstructed track (in these studies: true MonteCarlo track)

- look for not hit DOMs in close proximity of the track

likely candidate for a starting track

hit DOM  not hit DOM
finiteReco: Vertex Reconstruction

- Cherenkov light is being emitted under the characteristic angle $\theta \approx 41^\circ$
- calculate for each DOM the corresponding position on the track
finiteReco: Vertex Reconstruction

- Cherenkov light is being emitted under the characteristic angle $\theta \approx 41^\circ$
- calculate for each DOM the corresponding position on the track
- sort DOMs according to that position
- first hit defines reconstructed vertex

diagram:
1. Cherenkov light is emitted at $\theta \approx 41^\circ$.
2. Calculate the corresponding position on the track for each DOM.
3. Sort DOMs according to their position.
4. The first hit defines the reconstructed vertex.
recoVertex Distributions

signal

background

numu 2111 classicL1

reconstructed vertex x [m]
reconstructed vertex y [m]

# events [a.u.]

numu 2111 classicL1

reconstructed vertex x [m]
reconstructed vertex y [m]

# events [a.u.]

corsika 2130 classicL1

reconstructed vertex x [m]
reconstructed vertex y [m]

# events [a.u.]

corsika 2130 classicL1

reconstructed vertex x [m]
reconstructed vertex y [m]

# events [a.u.]

top view

side view
finiteReco: Hit Probabilities

- Cherenkov light is being emitted under the characteristic angle $\theta \approx 41^\circ$
- calculate for each DOM the corresponding position on the track
- look for not hit DOMs upstream of the vertex
- calculate hit probabilities for these DOMs

[Diagram with points representing DOMs and a reconstructed infinite track, showing first hit and not hit DOMs]
finiteReco: Likelihood Ratio

- calculate for each track likelihood ratio of two track hypotheses

\[
\log \frac{P(\text{noHit | Track})}{P(\text{noHit | noTrack})} =: \text{LLHR}
\]

probability for the observed hit pattern under the assumption of a \textit{through-going} track

probability for the observed hit pattern under the assumption of a \textit{starting} track

\[
P(\text{no Hit | Track}) \quad P(\text{no Hit | no Track})
\]

- hit DOM
- not hit DOM
Likelihood Ratio Distributions

- low LLHR value → track with high probability to start inside the detector

![Graph showing likelihood ratio distributions with tracks classified based on probability of starting inside the detector]
Likelihood Ratio Distributions

- low LLHR value → track with high probability to start inside the detector
Combined Efficiency

- apply cuts on the position of the reconstructed vertex and on the likelihood ratio
- signal: events starting inside DeepCore

background rejection and signal passing rate in MC data:

<table>
<thead>
<tr>
<th>cut level</th>
<th>background</th>
<th>signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>all triggers (L0)</td>
<td>1</td>
<td>--</td>
</tr>
<tr>
<td>DC trigger</td>
<td>$6 \times 10^{-2}$</td>
<td>100 %</td>
</tr>
<tr>
<td>velocity veto (L1)</td>
<td>$8 \times 10^{-3}$</td>
<td>99 %</td>
</tr>
<tr>
<td>finiteReco (L2)</td>
<td>$5 \times 10^{-7}$</td>
<td>36 %</td>
</tr>
</tbody>
</table>
Summary & Outlook

- DeepCore enables IceCube to look at the Southern Sky
- Large background has to be removed
- Use 2-stage veto process: velocity veto + finiteReco
- Rejection seems feasible

BUT

- Algorithm needs reconstructed track of high precision
- So far all studies only with MC truth

→ Reliable low energy track reconstruction needed!
→ Maybe use additional stages in between
Thank You!