Prepared for submission to JINST

Augmented Signal Processing in Liquid Argon Time Projection Chambers with a Deep Neural Network

H. W. Yu, [a,1] **M. Bishai,** [a] **W. Q. Gu,** [a] **M. F. Lin,**[b] **X. Qian,** [a] **Y. H. Ren,**[b] **A. Scarpelli,**[a] **B. Viren,** [a] **H. Y. Wei,**[a] **H. Z. Yu,** [c] **K. Yu,**[b] **C. Zhang**[a]

[a] *Physics Department, Brookhaven National Laboratory, Upton, NY, USA*
[b] *Computational Science Initiative, Brookhaven National Laboratory, Upton, NY, USA*
[c] *Sun Yat-Sen (Zhongshan) University, Guangzhou, China*

E-mail: hyu@bnl.gov

Abstract: The Liquid Argon Time Projection Chamber (LArTPC) is an advanced neutrino detector technology widely used in recent and upcoming accelerator neutrino experiments. It features a low energy threshold and high spatial resolution that allow for comprehensive reconstruction of event topologies. In current-generation LArTPCs, the recorded data consist of digitized waveforms on wires produced by induced signal on wires of drifting ionization electrons, which can also be viewed as two-dimensional (2D) (time versus wire) projection images of charged-particle trajectories. For such an imaging detector, one critical step is the signal processing that reconstructs the original charge projections from the recorded 2D images. For the first time, we introduce a deep neural network in LArTPC signal processing to improve the signal region of interest detection. By combining domain knowledge (e.g., matching information from multiple wire planes) and deep learning, this method shows significant improvements over traditional methods. This work details the method, software tools, and performance evaluated with realistic detector simulations.

Keywords: LArTPC, Signal Processing, Wire-Cell, Deep Neural Network

[1] Corresponding author

Contents

1 Introduction **1**

2 LArTPC Signal Processing **5**

3 Deep Neural Network ROI Detection **6**
- 3.1 Neural Network Architecture 6
- 3.2 Using Geometry Information 7
- 3.3 Simulation, Data Pre-processing, and Truth Label Preparation 8
- 3.4 Network Training 10

4 Performance and Discussions **11**

5 Summary **13**

A Fast Projection using the Ray Grid Technique **14**

1 Introduction

The Liquid Argon Time Projection Chamber (LArTPC) is a key detector technology for many current and anticipated future accelerator neutrino experiments [1–4]. When charged particles ~~transverse~~ traverse through the LAr medium, both scintillation light and ionization electrons are created. While the detection of scintillation light provides the event time, the detection of ionization electrons affords high-resolution position and energy information about particle trajectories. The rich event topology information provided by the LArTPC offers unique advantages in performing electron-photon separation. Together with superior calorimetry capability, the LArTPC is an excellent detector to study ν_μ to ν_e oscillations, which may hold the key to answering some remaining questions in the neutrino sector [5].

The current-generation LArTPCs are typically equipped with wire readouts, instead of pixelated readouts that is being developed rapidly [6], under the consideration of cost and heat production of electronics inside LAr [7]. In this configuration, the anode typically consists of multiple (commonly three) wire planes with different wire orientations. Under a uniform external electric field, the produced ionization electrons drift toward the anode planes at a (known) constant speed. The arrival time of ionization electrons, combined with the event time provided by the light detection system, allows for a position reconstruction of activities along the electric field direction. Figure 1 [1] illustrates the signal formation in a typical LArTPC configuration. The combination of wire signals from different planes provides the position information perpendicular to the electric field. Together, three-dimensional (3D) reconstruction of activities inside LArTPC can be made. To realize this concept, wire planes with different orientations must record signals from the same

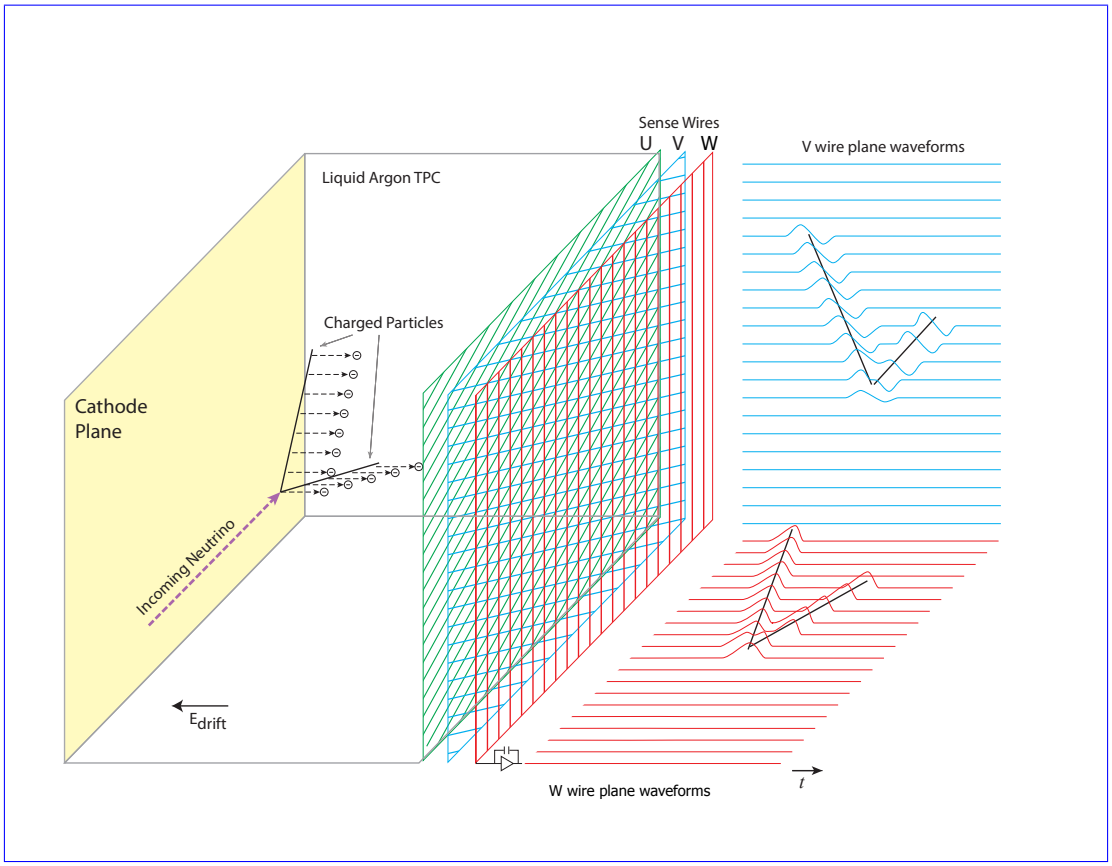

Figure 1: Diagram showing conceptual configuration of a typical three wire plane LArTPC and illustrating the signal formation of it. The signal in the U induction plane is omitted from the diagram for simplicity.

ionization electron cloud multiple times. This is achieved using both induction and collection wire planes. By properly configuring the electric fields between wire planes, the ionization electrons can fully pass through the first few (induction) wire planes then get collected by the last (collection) wire plane. When ionization electrons move close to the wires, induced ~~current~~ currents can be detected. As governed by Ramo's theorem [8], the induced current has bipolar and unipolar shapes on the induction-plane and collection-plane wires, respectively. While signal reconstruction for the collection-plane wires is generally simpler, it is more complex for the induction-plane wires given the ~~potential~~ potentially large cancellation effect of the bipolar signals. In this work, we introduce a novel signal processing procedure utilizing modern deep learning techniques and assisted by correlating information from multiple planes. This new procedure leads to an improved reconstruction of ionization electron distributions, which further enhances the quality of overall event reconstruction.

The recorded digitized TPC signal is a convolution of the distribution of the arriving ionization electrons and the impulse detector response:

$$M(t,x) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} R(t-t',x-x')\cdot S(t',x')dt'dx' + N(t,x), \tag{1.1}$$

where $M(t, x)$ is a measurement, such as an Analog to Digital Converter (ADC) value at a given sampling time and wire position; $R(t-t', x-x')$ is the impulse detector response, including both the field response that describes the induced current by a moving ionization electron and the electronics response from the shaping circuit; $S(t', x')$ is the charge distribution in time and space of the arriving ionization electrons; and $N(t, x)$ is the electronics noise. The goal of TPC signal processing is to reconstruct the original charge distribution S from the measurement M given the known detector response R in the presence of the electronics noise N.

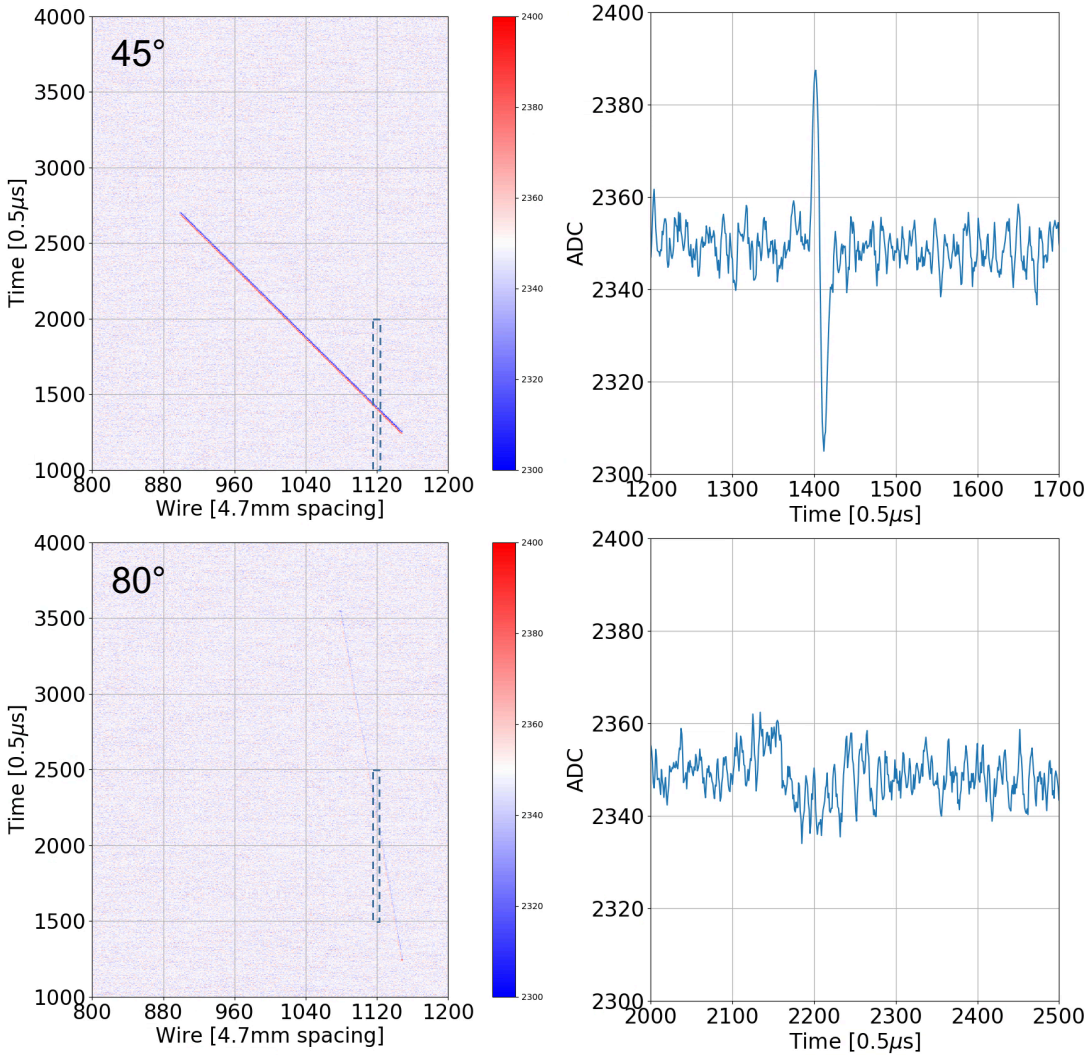

Figure 2: Left: projection of a minimum ionizing particle (MIP) track on one induction plane; right: a section of waveform of wire 1120 from the left plot indicated by the dashed box. Track angle projected on this plane is 45° (Top) and 80° (Bottom) with respect to the wire orientation. The recorded signal for the 80° track is weak as the result of the cancellation effect of the bipolar field response and the extended (in time) distribution of ionization electrons.

Developed from earlier work in ref. [9], the current state-of-the-art algorithm to reconstruct the original ionization electron distribution is the so-called "2D deconvolution" technique [10, 11]

using Discrete Fourier Transformation. This TPC signal processing algorithm is implemented in the *Wire-Cell* Toolkit [12] and can be run as a plug-in of the LArSoft software suite [13]. Although this algorithm has shown good performance and has been adopted in many experiments, there is still room to improve, especially for large angle tracks with respect to the wire orientation. These tracks are known as the "prolonged tracks" because they induce current on a given wire over an extended period of time [10]. Figure 2 illustrates the challenge in signal processing for prolonged tracks (with an 80°track angle). In this work, the track angle is defined by the track projection on a 2D plane, where the x-axis is the wire pitch direction and y-axis is the electron drifting direction. The track angle is measured by the angle between track projection and the x-axis in this plane, defined as θ_{xz}. This angle definition is the same as that used in figure 8 of ref. [10].

While the TPC signals for a track with a $\theta_{xz} = 45°$ are clear (see. figure 2), the ones for a track with $\theta_{xz} = 80°$ are rather weak. Such a big difference with respect to the track angle is the result of the bipolar and long-range nature of the induced current for a point source on the induction-plane wires. Thus a prolonged track leads to large cancellation of the induced current, resulting in a low ~~signal-to-background~~ signal-to-noise ratio. The same conclusion can be drawn when viewing from the frequency domain. First, the average response function of the induction-plane wires must approach to zero at low frequency (e.g., figure 12 of [10]) as the net charge collected on an induction-plane wire is zero [8]. Second, the signal of a prolonged track mostly resides at low frequency. Therefore, the measured signal strength, which is the product of signal strength from a prolonged track and the average response function, can become small compared with the low-frequency electronics noise, leading to a poor signal-to-noise ratio. To achieve good performance of the signal processing, ref. [10] introduced a region of interest (ROI) detection technique, whose goal is to define an ROI in the time domain to contain the distribution of ionization electrons. The ROI significantly reduces low-frequency noise for the induction planes, which, in turn, increases the signal-to-noise ratio. Obviously, to maximize the signal-to-noise ratio, the ROI fully containing the distribution of ionization electrons in the time domain should be as small as possible. For this reason, the ROI finding algorithm in ref. [10] worked on the deconvolved waveform instead of the raw waveform because the ROI found in the latter case is typically bigger as a consequence of the extended detector response. With a set of heuristic logic, the ROI searching algorithm in ref. [10] further utilizes the connectivity information of ROIs on adjacent wires in a wire plane to enhance ROI detection efficiency as the TPC signals tend to be continuous for any charged particle track. Of note, while the ROI detection is crucial for induction-plane signal processing, it is less important for collection-plane signals, where the field response is generally unipolar.

However, even with the usage of "2D deconvolution" technique and "connectivity" information, some TPC signals, e.g. prolonged tracks, are still hard to identify. Adding more information progressively into the heuristic logic of the ROI detection algorithm [10] is not a scalable approach. Hence, we propose a novel LArTPC signal processing procedure based on the Deep Neural Network (DNN). DNNs are known to manage complicated correlations and cover solution phase space more efficiently. In the field of high energy physics, there are many successful applications of DNNs (see refs. ~~[14–18]~~[14**? ?** –18], among others). Furthermore, inspired by the *Wire-Cell* imaging concept in ref. [19], the TPC signals from other wire planes can enhance ROI detection efficiency by using the geometry information from multiple wire planes. Combining machine learning and domain knowledge, the new DNN LArTPC signal processing shows significantly improved performance,

especially for prolonged tracks.

This paper is organized as follows: section 2 briefly reviews the current state-of-the-art signal processing, and section 3 introduces the new DNN signal processing. The DNN architecture, *Wire-Cell* imaging concept, simulation and data pre-processing, and network training are described in sections 3.1, 3.2, 3.3, and 3.4, respectively. The DNN signal processing performance evaluation is presented in section 4 followed by the section 5 summary.

2 LArTPC Signal Processing

As introduced in section 1, LArTPC signal processing extracts ionization charge distribution as a function of drift time and wire number from the induced current on the readout wires. Figure 3 shows a simplified flowchart of the state-of-the-art signal processing algorithm in Refs. [10, 11].

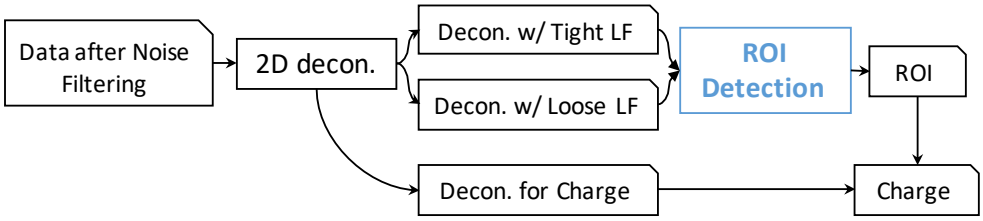

Figure 3: Flowchart of the LArTPC signal processing algorithm in ref. [10]. "LF" stands for low-frequency software filter, while "decon." denotes deconvolution.

Starting from the noise-filtered waveform [20], several 2D deconvolutions are performed with different software filters in the frequency domain. Each deconvolution follows Eq. 2.1, resolving the signal S using measurement M and detector response R in the frequency domain. A follow-up Inverse Fourier Transform (IFT) converts the signal S back to the time and space domain. Software filters (F) in the frequency domain are used to suppress noise at high and/or low frequencies. Here, the low-frequency filter is necessary (not used) to process signals in the induction-plane (collection-plane) wires as the bipolar nature of induction-plane wires' field response suppresses signals at low frequency. The different low-frequency filters have distinct features in the signal extraction. For example, a tight low-frequency filter limiting the bandwidth at a higher frequency value leads to a high-purity but low-efficiency signal extraction. On the other hand, a loose low-frequency filter expanding the bandwidth to lower frequency results in a low-purity but high-efficiency signal extraction.

$$S(\omega_t, \omega_x) \sim \frac{F(\omega_t, \omega_x) \cdot M(\omega_t, \omega_x)}{R(\omega_t, \omega_x)} \xrightarrow{IFT} S(t, x). \tag{2.1}$$

With the deconvolved waveform, ROI detection is performed to identify signal regions that are more likely caused by ionization signals (instead of noise). The ROI detection step intends to open windows just large enough to contain signals and maximize the signal-to-noise ratio. A set of heuristic logic is implemented to detect ROIs from the deconvolved signals with loose and tight low-frequency filters based on the connectivity information. With the ROI windows defined, the waveforms outside are suppressed to zero, which reduces data size significantly and makes

the reconstruction steps that follow computationally more efficient. The detected ROIs are then applied on the deconvolved waveform without applying low-frequency filters ("Decon. for Charge" in figure 3). While removing the low-frequency filters minimizes the distortion on the ionization electron distribution, applying ROIs in the time domain keeps the signal-to-noise ratio high, which is necessary for induction-plane wires.

The goal of ROI detection is to achieve high efficiency and high purity. This can be challenging when the signal-to-noise ratio is low, such as with the prolonged tracks described in section 1. In addition, the heuristic logic in ROI detection sometimes may fail in a busy situation where numerous activities are present (e.g., near neutrino interaction vertex or inside an electromagnetic shower). Section 3 introduces a novel ROI detection algorithm using a convolutional neural network based on information from multiple wire planes, which notably improves signal processing performance.

3 Deep Neural Network ROI Detection

In this section, we describe the details the Deep Neural Network (DNN) architecture, use of geometric information from multiple wire planes, the simulation and data pre-processing, and the network training.

3.1 Neural Network Architecture

To apply the DNN, the ROI detection problem is essentially labeling each pixel in a 2D image (with one dimension spanning LArTPC readout channels and the other drift time) as “signal” or “not-signal” (noise). Such a problem of dividing pixels from a picture into separated groups belongs to the class of machine learning procedures called *semantic segmentation*. We adopted the U-Net network architecture, introduced by O. Ronneberger et al. [21] in 2015, which is a type of convolutional network [**?**]. The U-Net has been developed into a family of network architectures (see Refs [17, 22], among others). Figure 4 shows the network configuration, which consists of an initial encoding path, a final decoding path, and skip connections which bridge these first two paths at various levels. Each level contains two 3×3 convolution layers and two batch normalization layers. In the encoding path, output tensors from each level are downsampled into the next level through a 2×2 max pooling operation. In the decoding path, output tensors from each level are upsampled by a bilinear upsample operation. The upsampled tensors are concatenated with tensors copied from the same level of the encoding path to form input for each level of the decoding path. The encoding path finds patterns at different levels based on the original information from input channels. In the decoding path, output channels are reconstructed from the patterns found from deeper levels or the skip connections. In the application of ROI detection, we use a five-level network. After a factor of 2 condensing in the height and width dimensions of each downsampling operation, in the deepest level of the network, pattern recognition is effectively performed in regions of $48 (= 3 \times 2^4) \times 48$ pixels.

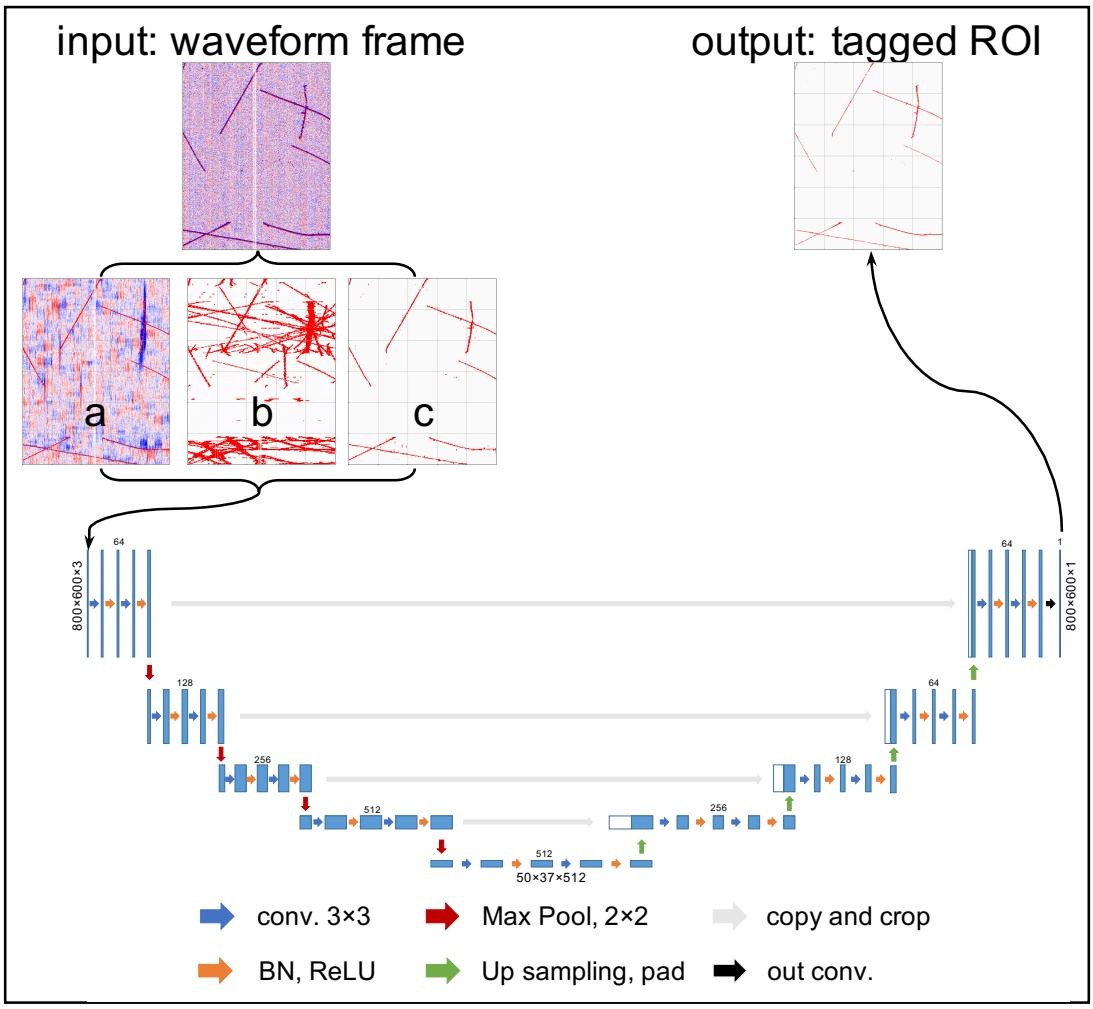

Figure 4: U-Net architecture used in the ROI detection for the DNN LArTPC signal processing. Intermediate 2D images are generated from original images containing the raw waveform. Several of these intermediate images are stacked to a multi-channel 2D image serving as the U-Net input. Numbers of intermediate images can vary. In this example, three are used: a) deconvolved signals from a loose low-frequency filter, b) MP2, and c) MP3. Output of the U-Net is a single channel 2D image labeling each pixel as signal (i.e., inside ROI) or not.

3.2 Using Geometry Information

In a LArTPC with multiple wire planes, each ionization electron is independently sensed by each wire plane. Therefore, information from the other wire planes can be used to assist ROI detection in the targeted wire plane by correlating the geometric relation among wires with different orientations. Such geometric constraints are expected to be more efficient for LArTPCs with three or more wire planes. For example, we use a three-plane configuration with the first two induction planes, labeled as "U" and "V", and the final collection plane, denoted as "W". This idea of *multi-plane constraint* in signal processing is inspired by the *Wire-Cell* tomographic reconstruction technique from ref. [19]. The procedure for implementing geometric constraints on the induction-plane wires is described

below and illustrated in figure 5:

1. For each channel, initial signal ROIs are formed by combining the deconvolved signals with tight and loose low-frequency software filters (see figure 3). The combination uses the connectivity information but skips the majority of heuristic logic in refining ROIs.

2. Across the channels, these initial signal ROIs are sliced into contemporaneous time slices of fixed duration (e.g., four time ticks or two μs). This choice ensures negligible loss of information following the Nyquist sampling theorem [**?**].

3. In a given time slice, we determine a subset of channels from each plane consisting of those that are inside the initial signal ROIs.

4. For each induction-plane channel inside the subset, we determine if any of its wires overlap with two wires from the other two subsets (one from each).* The successfully matched channels are referred to as the *three-plane coincidence* (MP3).

5. For each induction-plane channel outside of the subset, we determine if any of its wires overlap with two wires from the other two subsets (one from each). The successfully matched channels are referred to as the *two-plane coincidence* (MP2).

6. Steps 3 to 5 are repeated for every time slice.

Identifying the coincidence (MP2 or MP3) is a combinatorial problem with the potential to require prohibitive computational expense. To combat this, we developed an optimization technique for the primitive operations used to determine wire overlap by exploiting the symmetries of uniform wire direction and pitch. More details can be found in appendix A.

As shown in figure 4, three 2D images are used as input to the DNN to detect the ROI for each induction wire plane. The first image is the deconvolved result after applying the loose low-frequency software filter. The content in each 2D pixel (one channel and one time tick) is a float number representing the reconstructed ionization charge. The second image is the result of MP2. The content in each pixel is a number with the Boolean type and unity (zero) labeling the pixel to be inside (outside) the MP2. The last image is the result of MP3 also with Boolean type. Although MP2 and MP3 are obtained after rebinning four time ticks into one time slice, it is straightforward to determine if one pixel with one time tick is inside the MP2 or MP3 region. Figure 6 depicts an example of these arrays.

3.3 Simulation, Data Pre-processing, and Truth Label Preparation

The data samples used to train the network are generated from a comprehensive and detailed simulation over multiple stages of physics. These begin with an initial sampling of kinematics from cosmic ray muon models [23]; particle passage through detector material and the resulting energy depositions [24]; the production of electrons, including recombination effects [13]; and detector signal and noise simulation [12]. The detector response in the last step is provided by the *Wire-Cell* Toolkit that includes ionization electron drift, diffusion, stochastic fluctuations, field

*In the case of wrapped wires, one channel may correspond to multiple wires.

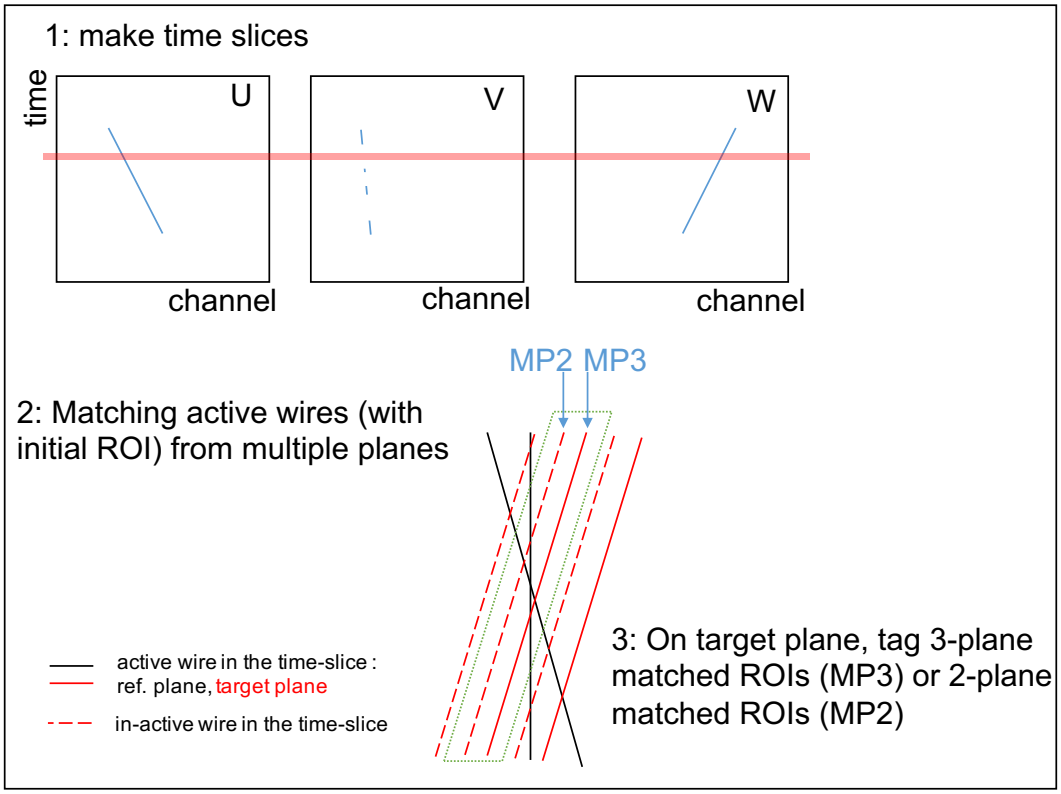

Figure 5: Illustration of creating MP2 and MP3 images. The application of geometric constraints is performed at each time slice (four time ticks).

response, and electronics response. The data pre-processing steps include excess noise filtering, initial deconvolution, and initial ROI finding. Excess noise filtering is particularly crucial in dealing with real detector data, but it is less important for simulation. The initial deconvolution and ROI finding are discussed in section 3.2 and more details can be found in ref. [25]. The simulation and data pre-processing chain is validated with the ProtoDUNE Single Phase detector (described in ref. [26]).

The input to DNN consists of MP2 (figure 6c), MP3 (figure 6d), and deconvolved signals from a loose low-frequency filter (figure 6b). Notably, MP2 and MP3 are generated from the deconvolved signals with tight (figure 6a) and loose low-frequency filters. For each event sample, there are a total of 6000 time ticks covering 3 ms drift time. The number of channels are 800, 800, and 960 for induction U, induction V, and collection W planes, respectively. The average number of cosmic tracks is about 11 per event. To limit memory use, 10 time ticks are rebinned into one time bin, which limits the typical data size of the induction plane to 600 time bins and 800 channels. For any 2D image, the content value in each rebinned pixel is taken as the average of the original value in 10 pixels (e.g., number of electrons for the deconvolved signal and 0 or 1 for MP2 and MP3). While the number of ticks used in this rebinning process could be further optimized, such a rebinning choice has minimal impact on the ROI detection because a majority of the ROIs (>95%) have a length longer than 10 time ticks. For network training purposes, the *truth* label is created by applying a simple ideal-detector model to the initial simulated ionization electron distributions (a smearing

with Gaussian distribution). A threshold is applied on the charge of a rebinned pixel (one channel and one time bin covering 10 time ticks) to determined if a rebinned pixel contributes to an ROI. In particular, the threshold is chosen to be 100 electrons on the average charge per time tick in a rebinned pixel. This value is much lower than the equivalent noise charge (ENC) from the entire electronics readout chain, which typically exceeds 400 electrons per time tick. The average number of (true) ROIs in an induction plane is about 3500 per event. Figure 6(e) shows an example of the truth label.

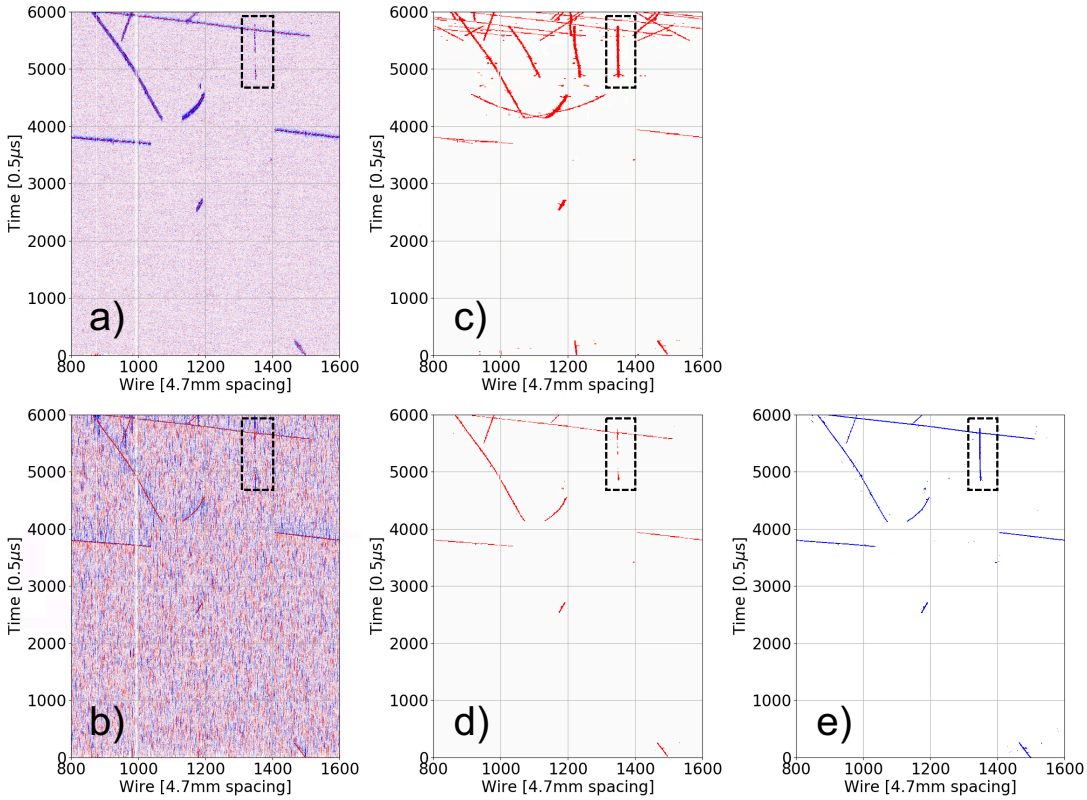

Figure 6: Example neural network input channels obtained from cosmic ray simulation for an induction wire plane: a) deconvolved image with a tight low-frequency filter; b) deconvolved image with a loose low frequency filter; c) multi-plane 2-plane coincidence (MP2); d) multi-plane 3-plane coincidence (MP3); and e) truth information. Here, the deconvolved images with tight and loose low-frequency filters from different wire planes are used to produce the MP2 and MP3 images. The displayed images feature original binning of 6000 time ticks and 800 channels. A prolonged track is indicated by the black rectangle dashed line. The input to DNN consists of b), c), and d).

3.4 Network Training

The network and supporting utilities are implemented using PyTorch [27] and are publicly accessible in ref. [28]. We trained the network on a platform with Intel i9-9900K CPU and NVIDIA 2080 Ti GPU with 11 GB VRAM [29]. The models used in the following paper are trained with

50 epochs. A training epoch consists of a total of 500 event samples (450 for training and 50 for validation) and requires six minutes on this platform. Each sample consists of one tensor holding the binary "signal"/"not-signal" truth label elements and a second tensor holding three 2D arrays (deconvolved signal with loose low-frequency filter, MP2, and MP3). With the cosmic ray simulation, each sample contains about 5 to 20 cosmic rays, resulting in an average of 3500 ROIs per sample. We use binary cross-entropy [30] as the loss function and stochastic gradient decent [31] with momentum 0.9 and learning rate 0.1 as the optimizer. Figure 7 shows the epoch mean loss as a function of the epoch number for both training and validation samples. We can see the two loss curves following a consistent trend and flattening out after about epoch 30. The absolute difference between the two curves are caused by the data sample difference between the training and validation datasets. In the next section, performance of this DNN ROI detection is evaluated.

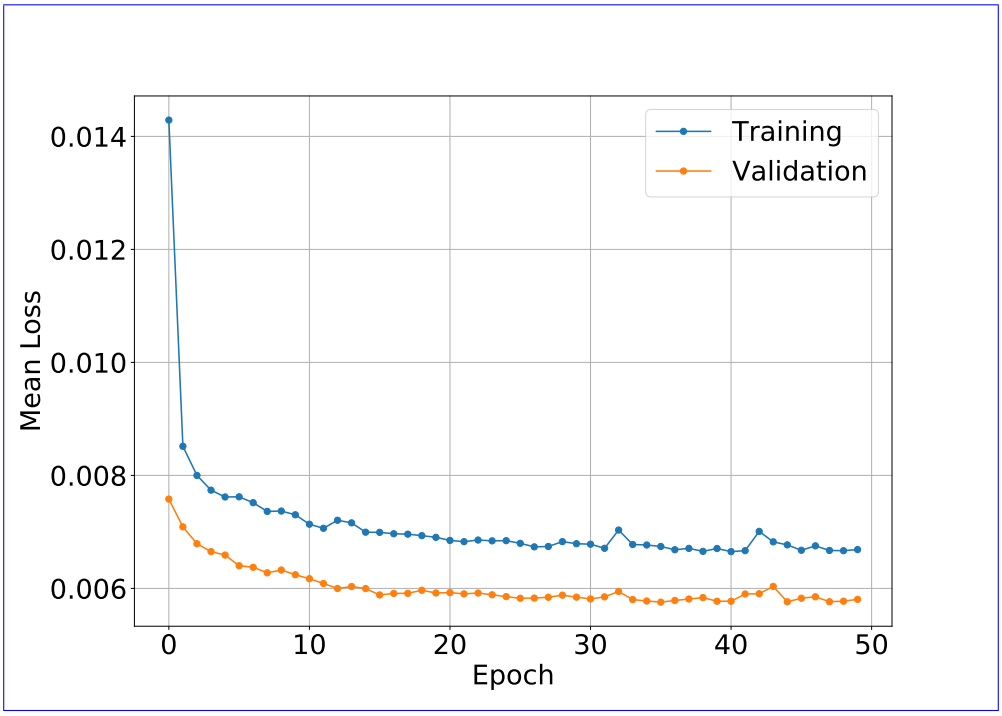

Figure 7: Epoch mean loss as a function of epoch number.

4 Performance and Discussions

The performance of the trained network with simulation is initially evaluated by focusing on the more challenging cases of the ideal (large-angle) prolonged tracks. For each 3D track, two projection angles, $\theta_{xz}(V)$ and $\theta_{xz}(U)$ with respect to V and U wire planes, are used to described its 3D direction. A larger projection angle with respect to a wire plane means a longer ROI in time, which is more challenging to detect. Two metrics are used to evaluate the performance with the rebinned pixels. Pixel-wise (rebinned) efficiency is defined as:

$$\text{Pixel Efficiency} := \frac{\text{\# of correctly label pixels}}{\text{\# of ROI pixels from truth label}}. \tag{4.1}$$

Pixel-wise purity is defined as:

$$\text{Pixel Purity} := \frac{\text{\# of correctly label pixels}}{\text{\# of ROI pixels from reconstruction}}. \tag{4.2}$$

In this evaluation, induction V wire plane is used as an example. Figure 8 compares pixel-wise ROI detection efficiency and purity for three algorithms: 1) current state-of-the-art ROI detection algorithm from ref. [25] as the reference, 2) DNN ROI detection without the multi-plane geometry information, and 3) DNN ROI detection with the multi-plane geometry information. Clear improvements can be gleaned from the DNN ROI detection algorithm:

1. The DNN ROI detection algorithm outperforms the reference algorithm, especially for extreme large-angle tracks ($> 85°$).

2. DNN ROI without the multi-plane information performs better than the reference but not as good as DNN ROI with the multi-plane information. In particular, by comparing the last three data points with 87° projection angle on the V plane (75°, 85°, and 87° projection angles on the U plane), we conclude that the multi-plane geometry information is more effective when the projection angles on the other wire planes are small (i.e., less prolonged). In the extreme case when the track travels completely perpendicular to the anode wire planes, the projection angles become 90° for both induction planes, and the multi-plane geometry information is less useful.

Using various, reconstructed event images, figure 9 shows additional DNN ROI detection algorithm improvements. Row (1) of figure 9 shows the reconstructed images of simulated ideal tracks with projection angles 87° on V plane and 85° on U plane for the three ROI detection algorithms (columns b-d), as well as the truth label (column a). Improvements in terms of efficiency and purity numbers (shown in figure 8) are evident. Rows (2) and (3) in figure 9 show reconstructed images from simulations generated from ~~charged pions overlaid with~~ cosmic rays. While row (2) focuses on the reconstruction of a prolonged cosmic ray track, row (3) centers on a busy interaction region of a charged pion. The DNN ROI detection algorithm with multi-plane information, column (d), has the best performance among all scenarios when compared with truth labels in column (a). Here we show the DNN ROI detection algorithm performance on the more challenging second induction plane (V plane) as an example, while the performance on the first induction plane (U plane) is similar. As for the collection plane (W plane), simple thresholding type ROI detection algorithm already produces robust results given the unipolar nature of the signals. Thus, we do dot use DNN ROI detection on the collection planes for now. The same conclusion can be drawn when applying the DNN ROI detection algorithm on the ProtoDUNE Single Phase data [32]. Preliminary results can be found in ref. [33].

In addition to U-Net, we also implemented and evaluated two other network structures: URestNet and Nested-U-Net. URestNet [17] is U-Net with residual connections. Nested-U-Net [22] is U-Net with dense skip connections. With proper optimizations in the training, all three networks perform similarly. U-Net uses slightly less (~10%) GPU memory in both training and inferencing.

For a production environment with mainly C++ software stack, we implemented application programming interfaces (APIs) to use TorchScript [34] in the *Wire-Cell* Toolkit [12]. The trained

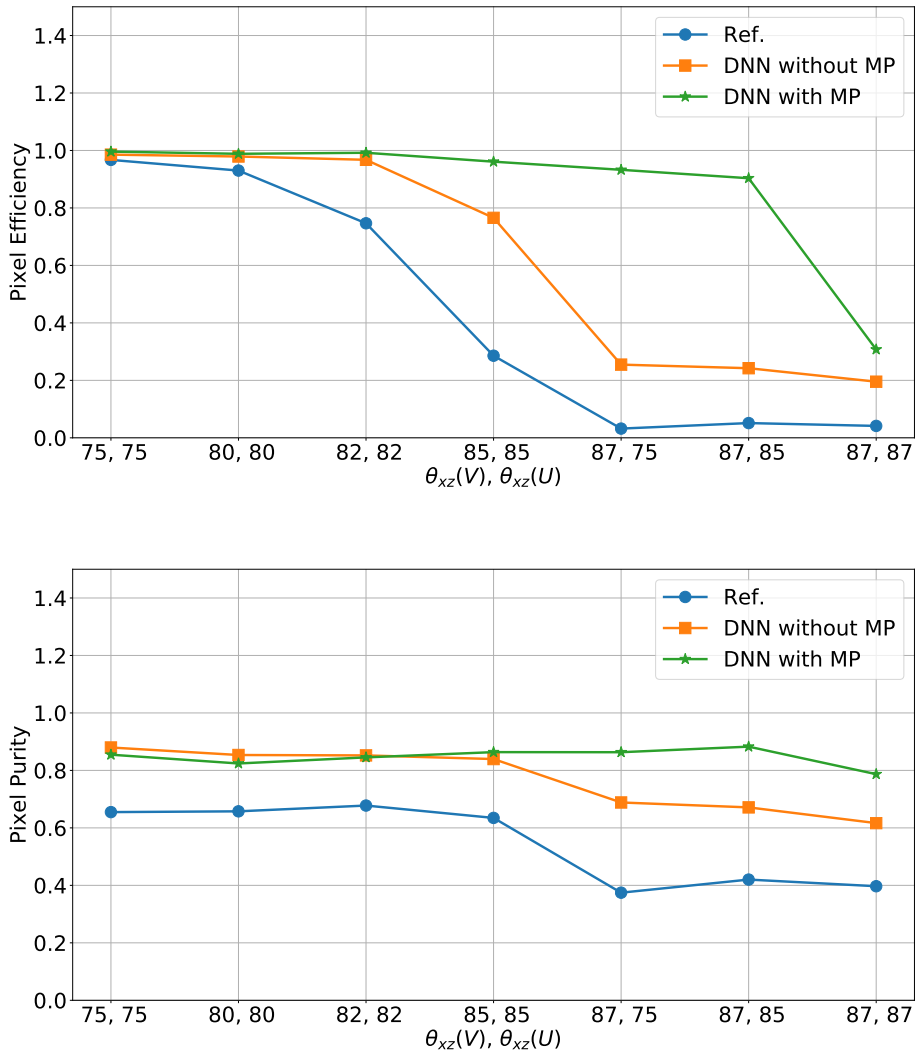

Figure 8: Pixel-wise (rebinned) ROI detection efficiency and purity for three ROI detection algorithms evaluated using simulated prolonged tracks. X-axis is track projection angle on the respective induction V and U planes in degrees. ROI detection is performed on the targeted induction V plane.

models are serialized in the TorchScript format. Table 1 shows a comparison of the inferencing time, memory usage, and VRAM use among the three ROI detection algorithms.

5 Summary

For the first time, we have developed a novel DNN ROI detection algorithm for LArTPC signal processing. As described in this paper, we adopted the ProtoDUNE configuration in the experiment, but the generic idea and procedure should work on all types of projective wire readouts. The implementation in the *Wire-Cell* Toolkit utilizes the TorchScript APIs, which enables using trained PyTorch models in the C++ production environment. This algorithm takes advantage of

Table 1: Comparison of resource usage for three ROI detection algorithms. Reference is the current ROI detection algorithm from ref. [25]; DNN CPU/GPU: DNN ROI detection algorithm proposed in this paper, inferencing with CPU/GPU, respectively.

Method	Time per plane [sec]	Memory [GB]	VRAM [GB]
Reference	0.40	1.3	-
DNN CPU	16.7	4.8	-
DNN GPU	0.14	3.7	3.7

modern machine learning techniques, as well as the domain knowledge specific to LArTPC signal processing (2-D deconvolution, multi-plane geometry correlation, etc.). Quantitative evaluations show this DNN ROI detection algorithm outperforms the current state-of-the-art ROI detection algorithm. Such improvement is expected to generate a better reconstruction of event topology in a LArTPC, particularly for detectors with higher noise and lower signal-to-noise ratio (e.g., with warm electronics [3]).

Acknowledgments

This work is supported by the U.S. Department of Energy, Office of Science, Office of High Energy Physics and Early Career Research Program under contract number DE-SC0012704. We thank the DUNE collaboration for use of its Monte Carlo simulation software and related tools. We would also like to show our gratitude to Charity Plata (Computational Science Initiative of Brookhaven National Laboratory) for her inputs during the manuscript preparation.

A Fast Projection using the Ray Grid Technique

Central to the use of geometry information for improving signal processing efficiency, as well as in the *Wire-Cell* 3D tomographic reconstruction technique [19], are a number of primitive operations that must be carried out numerous times by these higher-level algorithms. Optimization of these frequently called operations is critical for overall speed. Two core and related operations involve locating the intersection point of a wire from each plane and then locating a wire from a third plane relative to this intersection. The so-called *ray grid* method is used to optimize these operations by exploiting two symmetries of an idealized wire plane: fixed wire pitch and direction. Exploiting these symmetries transforms the problem from solving a vector equation to a simpler problem of applying indices.

With these symmetries assumed, we start by generalizing a fixed number of parallel wires in a wire plane to an infinite number of parallel *rays* in a *layer*. This generalization allows us to define the active area of an anode by introducing two logical layers, one providing horizontal and the other providing vertical boundaries via rays with pitch equal to the active height and width, respectively. The higher-level algorithms do not care about wires per se but about the logical lines running parallel to and halfway between two neighboring wires and in so in practice we associate the abstract rays to these lines. Finally, as the higher-level algorithms work in a projected space, we assume all layers are co-planar, which allows the problem to reduce to two dimensions.

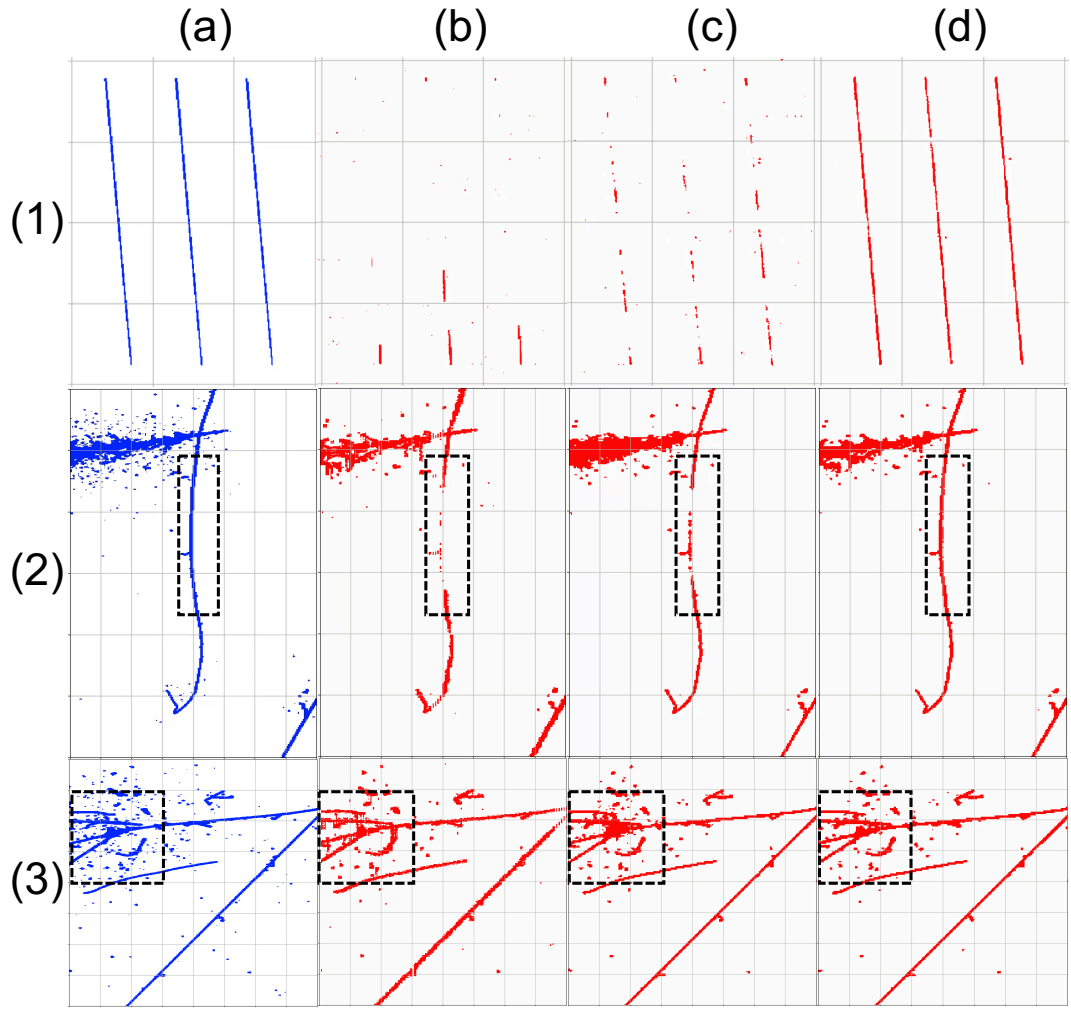

Figure 9: Event display showing the detected ROIs on different types of event topologies: (1) Ideal straight-line prolonged tracks with track angle of 87° projected to V plane and 85° projected to U plane, (2) cosmic track with a section featuring a large projection angle (indicated by the black rectangle dashed line), and (3) an interaction vertex of a charged pion with many activities (indicated by the black rectangle dashed line). ROI detection is performed on the V plane (target plane). The column labels are: (a) truth label, (b) reference ROI detection algorithm [25], (c) DNN ROI detection algorithm without the multi-plane geometry information, and (d) DNN ROI detection algorithm with the multi-plane geometry information.

A set of rays in a layer and their relationship to rays in a second or third layer may be categorized by a number of tensors listed below and illustrated for a three layer problem in figure 10. Within one layer we may consecutively number the rays and we mark these tensor indices with subscripts ($\in \{i, j, k\}$). We identify a special ray with an index of zero. The precise identification does not matter for the *ray grid* method. In practice the ray nearest to an edge of the active region is chosen. For quantities that span multiple layers, we mark them with superscript layer indices ($\in \{l, m, n\}$).

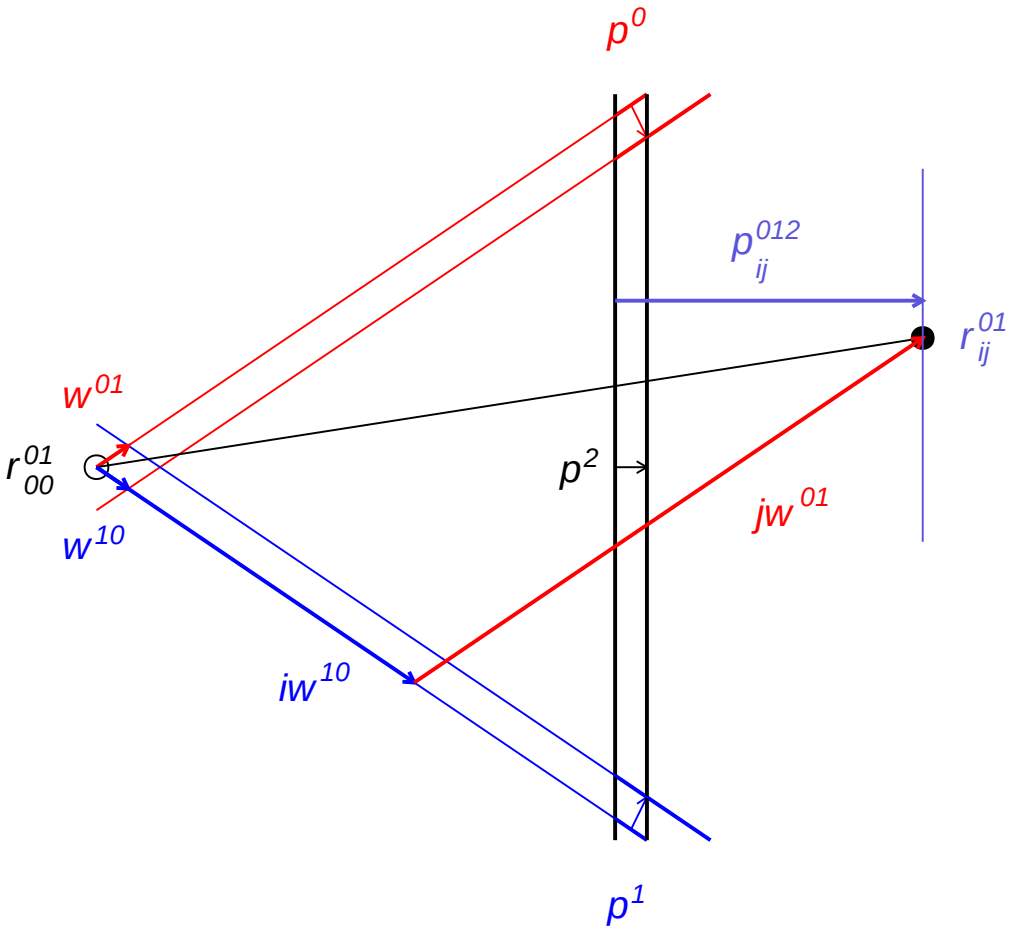

Figure 10: Example points, vectors and tensors involved in constructing and using ray grids. Layers 0, 1 and 2 are represented in red, blue and black, respectively. See text for further definitions.

p^l the pitch vector for layer l gives the displacement perpendicular to the layer's rays and has magnitude that of the pitch separation of two neighboring rays.

c^n the origin vector for layer n locates the center point of the specially identified ray $i = 0$.

r^{lm}_{ij} the crossing point of ray i from layer l and ray j from layer m.

r^{lm}_{00} the crossing point the specially identified zero rays of layers l and m.

w^{lm} a vector giving the displacement along the direction of a ray in layer l between the crossing points of that ray and two neighboring rays from layer m.

The last two tensors, r^{lm}_{00} and w^{lm} can be calculated with simple vector arithmetic in a pair-wise manner among the N_l layers in the ray grid. The former is symmetric and both have undefined diagonals. These do require $O(N_l^2)$ operations where N_l is the number of layers and typically $N_l = 5$. Given these two tensors, arbitrary crossing points of rays from two different layers can be written as,

$$r_{ij}^{lm} = r_{00}^{lm} + jw^{lm} + iw^{ml}. \tag{A.1}$$

By exploiting the constant ray direction and pitch, this tensor replaces numerous calls to $sin()$, $cos()$ and $sqrt()$ functions with multiplication, addition and array indexing.

The location of a crossing point r_{ij}^{lm} along the pitch direction of a third layer n may be written as,

$$p_{ij}^{lmn} = (r_{ij}^{lm} - c^n) \cdot \hat{p}^n. \tag{A.2}$$

Expanding this in terms of r_{ij}^{lm} shows the number of unique vector operations needed to form this tensor is combinitoric in the number of layers (eg, $N = 5$) and independent from the number of wires:

$$P_{ij}^{lmn} = r_{00}^{lm} \cdot \hat{p}^n + jw^{lm} \cdot \hat{p}^n + iw^{ml} \cdot \hat{p}^n - c^n \cdot \hat{p}^n. \tag{A.3}$$

Finally, the typical use of this pitch location is to identify the nearest ray in plane n by its index. This index can be found simply as,

$$I_{ij}^{lmn} = floor(P_{ij}^{lmn}/p^n). \tag{A.4}$$

References

[1] R. Acciarri et al. Design and Construction of the MicroBooNE Detector. *JINST*, 12(02):P02017, 2017, 1612.05824.

[2] B. Abi et al. The Single-Phase ProtoDUNE Technical Design Report. 2017, 1706.07081.

[3] M. Antonello et al. A Proposal for a Three Detector Short-Baseline Neutrino Oscillation Program in the Fermilab Booster Neutrino Beam. *arXiv:1503.01520*, 2015, 1503.01520.

[4] Babak Abi et al. Deep Underground Neutrino Experiment (DUNE), Far Detector Technical Design Report, Volume II DUNE Physics. 2 2020, 2002.03005.

[5] M. V. Diwan, V. Galymov, X. Qian, and A. Rubbia. Long-Baseline Neutrino Experiments. *Ann. Rev. Nucl. Part. Sci.*, 66:47–71, 2016, 1608.06237.

[6] D.A. Dwyer et al. LArPix: Demonstration of low-power 3D pixelated charge readout for liquid argon time projection chambers. *JINST*, 13(10):P10007, 2018, 1808.02969.

[7] Veljko Radeka et al. Cold electronics for 'Giant' Liquid Argon Time Projection Chambers. *J. Phys. Conf. Ser.*, 308:012021, 2011.

[8] S. Ramo. Currents induced by electron motion. *Proceedings of the IRE*, 27:584, 1939.

[9] Bruce Baller. Liquid argon TPC signal formation, signal processing and reconstruction techniques. *JINST*, 12(07):P07010, 2017, 1703.04024.

[10] C. Adams et al. Ionization electron signal processing in single phase LArTPCs. Part I. Algorithm Description and quantitative evaluation with MicroBooNE simulation. *JINST*, 13(07):P07006, 2018, 1802.08709.

[11] C. Adams et al. Ionization electron signal processing in single phase LArTPCs. Part II. Data/simulation comparison and performance in MicroBooNE. *JINST*, 13(07):P07007, 2018, 1804.02583.

[12] https://github.com/WireCell/wire-cell-toolkit.

[13] http://larsoft.org/.

[14] A.M. Sirunyan et al. Identification of heavy-flavour jets with the CMS detector in pp collisions at 13 TeV. *JINST*, 13(05):P05011, 2018, 1712.07158.

[15] Chase Shimmin, Peter Sadowski, Pierre Baldi, Edison Weik, Daniel Whiteson, Edward Goul, and Andreas SÃÿgaard. Decorrelated Jet Substructure Tagging using Adversarial Neural Networks. *Phys. Rev. D*, 96(7):074034, 2017, 1703.03507.

[16] Peter Sadowski, Balint Radics, Ananya, Yasunori Yamazaki, and Pierre Baldi. Efficient antihydrogen detection in antimatter physics by deep learning. *J. Phys. Comm.*, 1(2):025001, 2017, 1706.01826.

[17] C. Adams et al. Deep neural network for pixel-level electromagnetic particle identification in the MicroBooNE liquid argon time projection chamber. *Phys. Rev. D*, 99:092001, May 2019.

[18] Pierre Baldi, Jianming Bian, Lars Hertel, and Lingge Li. Improved Energy Reconstruction in NOvA with Regression Convolutional Neural Networks. *Phys. Rev. D*, 99(1):012011, 2019, 1811.04557.

[19] Xin Qian, Chao Zhang, Brett Viren, and Milind Diwan. Three-dimensional Imaging for Large LArTPCs. *JINST*, 13(05):P05032, 2018, 1803.04850.

[20] R. Acciarri et al. Noise Characterization and Filtering in the MicroBooNE Liquid Argon TPC. *JINST*, 12:P08003, 2017, 1705.07341.

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[22] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: Redesigning skip connections to exploit multiscale features in image segmentation, 2019, 1912.05074.

[23] Ralph Engel, Dieter Heck, Tim Huege, Tanguy Pierog, Maximilian Reininghaus, Felix Riehn, Ralf Ulrich, Michael Unger, and Darko Veberič c. Towards a Next Generation of CORSIKA: A Framework for the Simulation of Particle Cascades in Astroparticle Physics. *Comput. Softw. Big Sci.*, 3(1):2, 2019, 1808.08226.

[24] S. Agostinelli et al. Geant4 – a simulation toolkit. *Nucl. Instrum. Meth.*, A506:250–303, 2003.

[25] C. Adams, R. An, J. Anthony, J. Asaadi, M. Auger, L. Bagby, S. Balasubramanian, B. Baller, C. Barnes, G. Barr, and et al. Ionization electron signal processing in single phase lartpcs. part i. algorithm description and quantitative evaluation with microboone simulation. *Journal of Instrumentation*, 13(07):P07006âĂŞP07006, Jul 2018.

[26] B. Abi et al. First results on ProtoDUNE-SP liquid argon time projection chamber performance from a beam test at the CERN Neutrino Platform. 7 2020, 2007.06722.

[27] https://pytorch.org.

[28] Haiwang Yu. Pytorch-unet for lartpc signal processing. https://github.com/HaiwangYu/Pytorch-UNet.

[29] https://www.nvidia.com/en-us/geforce/graphics-cards/rtx-2080-ti/.

[30] Binary Cross Entropy between the target and the output in PyTorch https://pytorch.org/docs/master/generated/torch.nn.BCELoss.html.

[31] Stochastic gradient descent (optionally with momentum) in PyTorch. https://pytorch.org/docs/stable/optim.html#torch.optim.SGD.

[32] B. Abi et al. The Single-Phase ProtoDUNE Technical Design Report. 6 2017, 1706.07081.

[33] Haiwang Yu. Deep neural network signal processing for liquid argon time projection chamber. http://meetings.aps.org/Meeting/APR20/Session/X13.1.

[34] https://pytorch.org/docs/stable/jit.html.