Wire Cell Toolkit Status

With Focus On LArSoft Integration

Brett Viren

(for WCT team)

Physics Department

BROOKHAVEN NATIONAL LABORATORY

μB Ana Tools

2017 Aug 24

Outline

Toolkit Overview & Recent Improvements

Wire Cell – LArSoft Integration Design

Noise Filter + Signal Processing

Memory Issue

To Do & Summary

Overview of Wire Cell Toolkit

The toolkit consists of *software components* covering:

- Full **signal simulation** using **long-range and fine-grained detector response** and accurate **electronics response** functions.
- Full **noise simulation** using realistic, empirical and first-principles models.
- **Noise filter** to reduce (μBoone's) excess noise.
- **Signal processing** to deconvolve **detector response** and select **signal-ROI**.
  - Charge matching on all 3 planes, confirmed with μBooNE data.
  - Evaluated against full simulation to remove biases.

On the the way from Wire Cell **Prototype**:

- 3D ionization charge image reconstruct, pattern recognition, optical, ...

Recent Toolkit Improvements

Since last meeting, many important improvements to WCT:

- Prototype **signal processing** ported into toolkit (Xin, bv)
- Improved and validated signal processing to greatly **reduce charge bias** as a function of track angle (Hanyu)
  - → Now feel WCT sigproc gives solid foundation for 3D imaging.
- Refinement of sim's **signal induction** model (Hanyu)
- Corrections to **electronics response** model (Hanyu)
- Initial plan and steps for **simulation truth** recording (Brooke)
- Modeling and validating **2D field calculations** of detector pathology regions (Yichen, Mike).
- Start of a **validation suite** (bv)
- Improved **LArSoft integration** (bv) ... focus of this talk

Toolkit Overview & Recent Improvements

Wire Cell – LArSoft Integration Design

Noise Filter + Signal Processing

Memory Issue

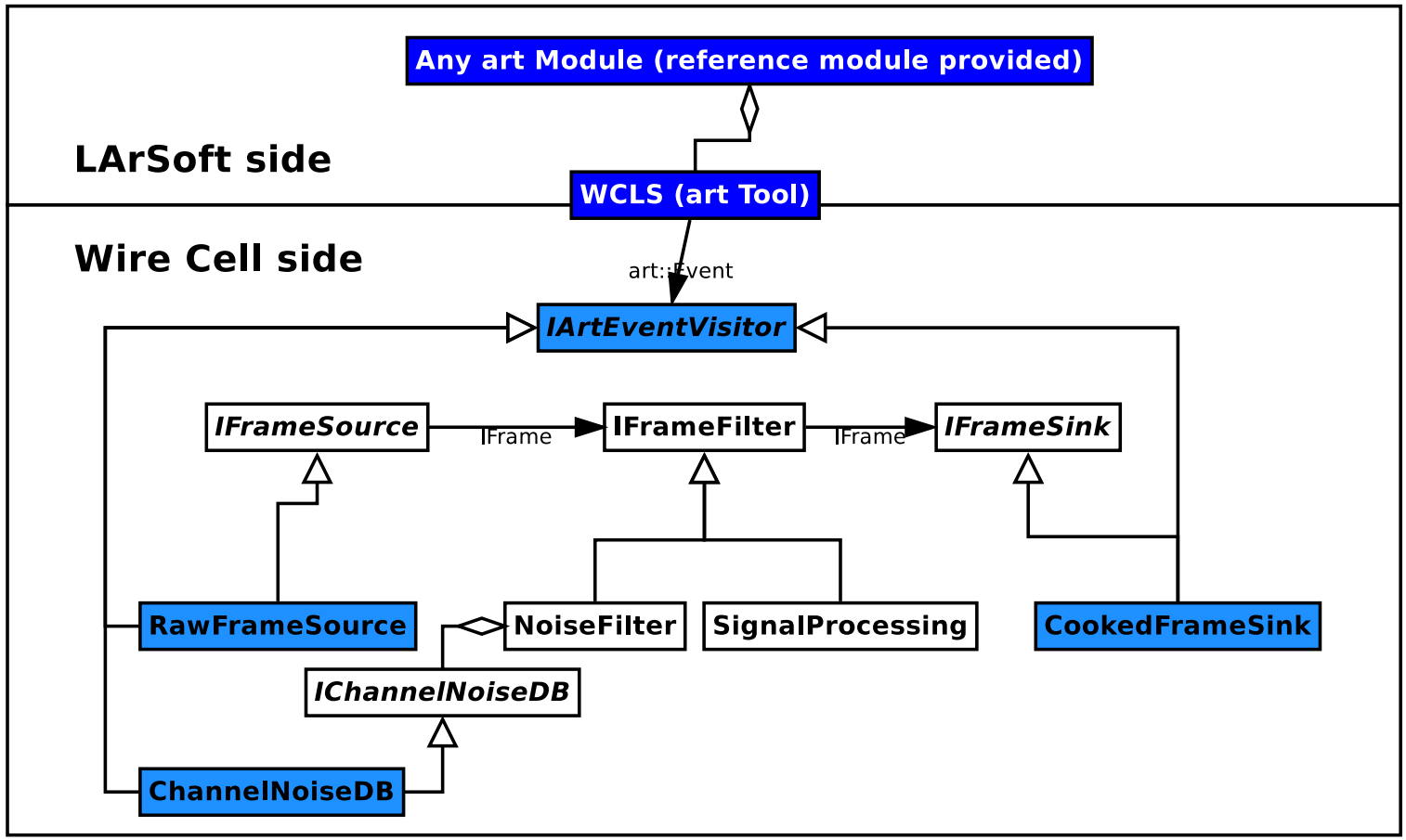
To Do & Summary

Initial Noise Filter Integration

- Dedicated MicroBooNE **noise filter** (Brian).
  - Got something working quickly, minimizing up-front effort.
  - It is kept as-is in `larwirecell/LarInterfaces`
- This helped to identify some issues:
  - Code implements a **tight coupling** between LS and WCT sides.
  - Requires detailed development updates as LS or WCT sides change.
  - Extending this pattern into the future will require a lot of coding, is error prone and tends toward copy-paste programming.
  - Lack of clear boundary leads to difficulty capturing full code, data and configuration state from LS-side for WCT-side debugging.

$\rightarrow$ The new integration design addresses these issues.

New WCT/LS Integration Design



art components | converter classes | **WCT classes**.

Blue parts live in integration package `larwirecell` branch `feature/integration`.
Produces WCT plugin `WireCellLarsoft` + *art* libraries + FHiCL files.

Words for last slide

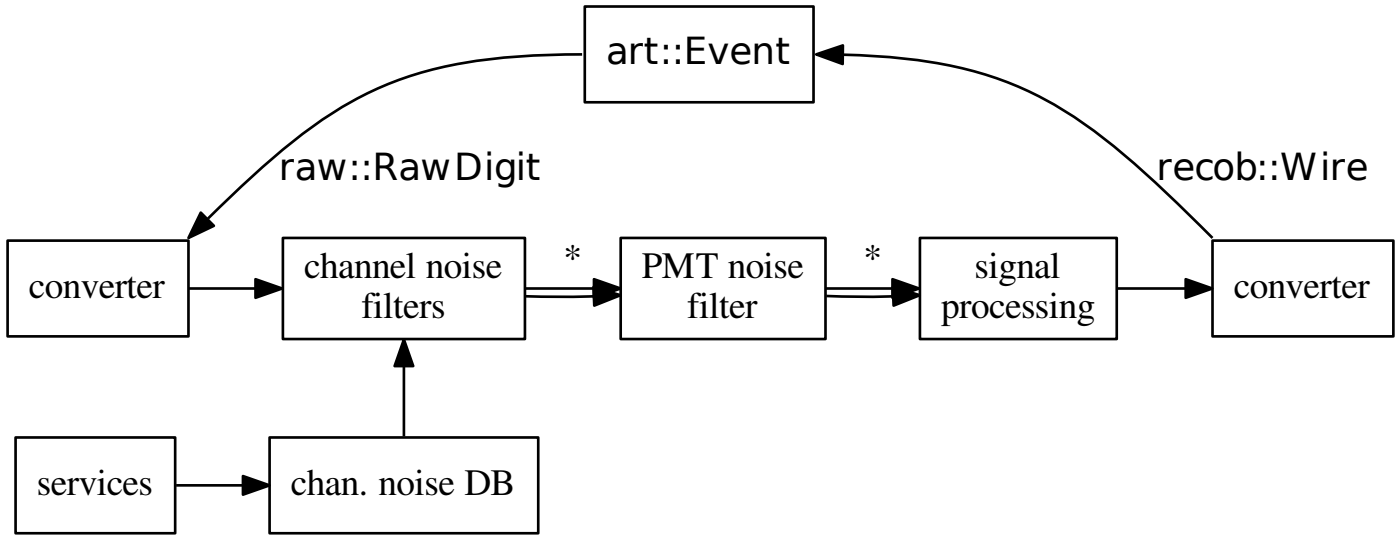
- Any *art* Module can be used, the `WireCellToolkit_module` provided as reference (likely only one needed).
- Main WCT entry, the *art* Tool `WCLS` provides functionality equivalent to the `wire-cell` command line interface program.
  - Handles WCT component construction, configuration, aggregation and execution.
- `IArtEventVisitor` is the main WCT *component interface* to *art* and is the basis of **converter components**.
  - `RawFrameSource`: `raw::RawDigit` → `IFrame`
  - `CookedFrameSink`: `IFrame` → `recob::Wire`
  - `ChannelNoiseDB`: feed time-varying info to noise filter.
- Core WCT code shown is NF+SP, but any WCT job fits this pattern.
  - → Support for new WCT functionality only requires new *converter components* to be written, *iff* new data products are introduced.
  - → WCT code run in LS can be rerun, largely unchanged, in a pure-WCT environment which help experts to debug problems.

Configuration: *art*-side FHiCL

```
source :{
   module_type: RootInput
   inputCommands: ["drop *", "keep raw::RawDigits_*_*_*"]
}
physics :{ producers: { nfsp : {
  module_type : WireCellToolkit
  wcls_main: {
    tool_type: WCLS
    plugins: ["WireCellGen", "WireCellSigProc", "WireCellLarsoft"]
    apps: ["Omnibus"]
    inputers: ["wclsRawFrameSource", "wclsChannelNoiseDB"]
    outputers: ["wclsCookedFrameSink"]
    configs: ["uboone-nfsp.jsonnet"]
    params: { detector: "uboone" }
   }
   p1 : [ nfsp ]
   trigger_paths : [ p1 ]
}
```

The `WCLS/wcls_main` *art* Tool accepts FHiCL parameter set which is essentially identical the arguments that the "`wire-cell`" CLI takes.

NF+SP Job Overview



- Currently NF+SP are combined with no intermediaries saved to `art::Event`.
  - In addition to waveforms, NF produces "bad regions" in channel vs tick space which must be communicated to SP.
- Determine "bad region" data product ⇒ can save NF output to `art::Event`.

NF+SP Knobs

A large amount of options can be set in WCT configuration.

- Nominal and per channel field responses, gain, shaping.
- Per-chan gain/shaping to reconfigure misconfig channels.
- Frequency masks, electronics parameters.
- High/low pass signal processing filters.
- Detailed algorithm parameters (thresholds, windows, paddings, offsets).

→ Configuration uses simple but powerful data language Jsonnet. Large config data sets (responses, wires) provided as (`bzip2`'ed) JSON.

→ Production configuration can be transplanted largely as-is to "pure" WCT environment for detailed deugging.

Memory Issue (not a leak!)

WCT `ExecMon` memory footprints (`VmSize`)		
1.5 GB	pure WCT job[1]	no output file
2.0 GB	pure WCT job	simple TH2 output
3.0 GB	WCT + LS	no output file
4.5 GB	WCT + LS	*art* output file

- Naively, memory footprint "should" be no more than 1 GB
  - 2 $\times$ 300MB μB frames + $\sim$200MB config+lib data
- *art*'s `VmPeak` and `VmHWM` report slightly higher than table
  - last row $\rightarrow$ `VmPeak = 5298.59 VmHWM = 4881`
- **Chris Backhouse**, "set `saveMemoryObjectThreshold` 100 MB"
  - last row $\rightarrow$ `VmPeak = 3810.77 VmHWM = 3393.08`

What do?

- Suspect **memory fragmentation** is at fault.
- Maybe mitigate WCT's contribution by preallocating output memory.
  $\rightarrow$ but that leaves ROOT's and/or *art*/LArSoft's contribution.
- Other ideas? For now, I leave it but I wanted people to be aware.

[1] The "pure WCT job" = event, "WCT+LS" = multiple events.

Toolkit Overview & Recent Improvements

Wire Cell – LArSoft Integration Design

Noise Filter + Signal Processing

Memory Issue

To Do & Summary

WCT Release Roadmap

0.5.2 Support for μB NF module (past release).

0.6.x Support for NF+SP with new LS integration design.

- In progress now.
- Will also merge in `larwirecell`'s `feature/integration` branch for subsequent LS release.

0.7.x Support for saving sim truth and multi-field response (eg, for "touching wires" in MB).

- Followed by `larwirecell` update with needed converters.

0.8.x 3D imaging in WCT.

Caveat: exact version strings may differ and priorities may change.

Release 0.6.0 work

1. Final validation going on now. (WCT team)
2. WCT's own git-gymnastics for release. (bv)
3. Create new and update existing FNAL UPS product "shim" builder scripts in coordination with Lynn. (bv)
4. Make local UPS product build of `wirecell`, `larwirecell` and new external versions. (bv)
5. Merge `larwirecell` branch and do official builds. (Lynn)
6. Invite testing by μBoone experts: (all)
   - Check if established assumptions about output `recob::Wire` are met?
7. Iterate to 0.6.1+
   - Hopefully not needed!
8. Integrate into μBoone production processing.

Summary

- Many improvements and refinements in WCT.
  - → **"Ready for 3D imaging!"**
- **WCT/LS integration is a solved problem**
  - All issues are at least understood.
  - NF+SP integration provides examples for future work.
  - New WCT functionality now easy to integrate and maintain.
  - New data converters only new integration coding needed.
- NF+SP release in progress.
  - Then, sim. Then 3D imaging.
- Memory inflation issue may be a (broader?) concern.

fin