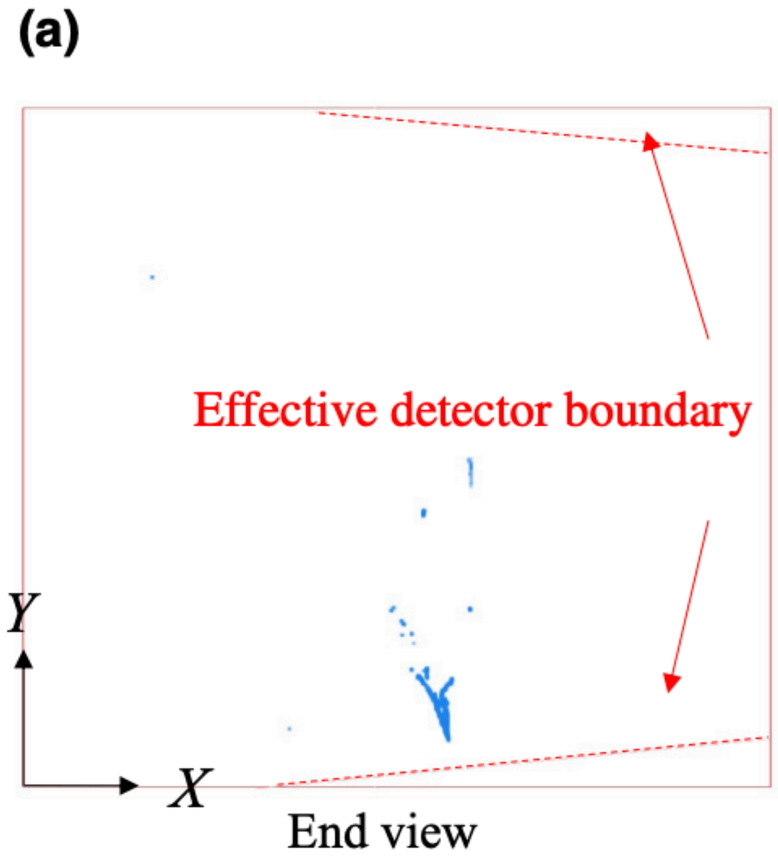
SCE Correction for SBND WirecellI

Avinay Bhat, Lynn Tung, Haiwang Yu
04/23/2026

Why SCE matters for Wire-Cell?

- For single-APA clustering, raw treatment may be acceptable as a first approximation.
- For multi-APA stitching, SCE distorts positions and affects crossing-track continuity.
- For cosmic tagging and fiducial logic, SCE changes the corrected detector boundaries.
- Therefore, the most important near-term Wire-Cell use cases are:
  - APA stitching with crossing muons
  - cosmic tagger / corrected fiducial volume logic



What we learned from Lane’s work?

- Historically, one TH3 map set served both drift volumes.
- Lane’s update introduced dual maps, one for East and one for West drift volumes.
- The relevant SBND data products now include dual-map ROOT files such as: SCEoffsets_SBND_E500_dualmap_CV_voxelTH3.root
- For reconstruction-style correction, we want the backward calibration displacement maps, not the forward simulation maps.

WCT and sbndcode versions

- SBND software base
  - sbndcode v10_14_02_02
  - Local MRB development area built from this release
  - Local larwirecell v10_01_28 in the MRB area

- Wire-Cell Toolkit
  - Base toolkit matched to wirecell v0_33_0
  - Local patched WCT built from the 0.33.0 source tag
  - SBND runtime confirmed to load the patched local WCT libraries

Where the change belongs in Wire-Cell?

- Wire-Cell insertion points
- Code side:
  clus/src/PCTransforms.cxx
- Config side:
  pgrapher/common/clus.jsonnet
- Relevant existing hook:
  switch_scope(name="",
  correction_name="T0Correction")

```
#include <map>
#include <memory>
#include <string>

class PCTransformSet;

WIRECELL_FACTORY(PCTransformSet, PCTransformSet,
                 WireCell::Clus::IPCTransformSet,
                 WireCell::IConfigurable)


// Note, we do not register any of the individual IPCTransforms because the
// crazy clus code evolved to reinvent this pattern and it's too damn ugly at
// this point to try to refactor properly.


using namespace WireCell;
using namespace WireCell::Clus;

class T0Correction : public WireCell::Clus::IPCTransform
{
public:

    virtual ~T0Correction() = default;

    T0Correction(IDetectorVolumes::pointer dv)
        : m_dv(dv) {

        for (const auto& [wfid, _] : m_dv->wpident_faces()) {
            WirePlaneId wpid(wfid);
            m_time_global_offsets[wpid.apa()][wpid.face()] = m_dv->metadata(wpid)["time_offset"].asDouble();
            m_drift_speeds[wpid.apa()][wpid.face()] = m_dv->metadata(wpid)["drift_speed"].asDouble();
        }
    }

    /**
     * From time2drift in Facade_Util.cxx
```

Implemented transforms in PCTransforms.cxx

- Added:
  - SCECorrection
  - T0SCECorrection

- Transform logic:
  - SCECorrection: applies SBND backward displacement maps point-by-point
  - T0SCECorrection: applies T0Correction first, then SCECorrection

- ROOT inputs used:
  - TrueBkwd_Displacement_X_E/Y_E/Z_E
  - TrueBkwd_Displacement_X_W/Y_W/Z_W

```
class T0SCECorrection : public WireCell::Clus::IPCTransform
{
public:
  virtual ~T0SCECorrection() = default;

  T0SCECorrection(IDetectorVolumes::pointer dv,
                  const std::string& sce_file,
                  double cathode_eps = 2.5)
    : m_t0(dv)
    , m_sce(dv, sce_file, cathode_eps)
  {}

  virtual Point forward(const Point& pos_in, double cluster_t0, int face, int apa) const override
  {
    auto p1 = m_t0.forward(pos_in, cluster_t0, face, apa);
    return m_sce.forward(p1, cluster_t0, face, apa);
  }

  virtual Point backward(const Point& pos_in, double cluster_t0, int face, int apa) const override
  {
    auto p1 = m_sce.backward(pos_in, cluster_t0, face, apa);
    return m_t0.backward(p1, cluster_t0, face, apa);
  }

  virtual bool filter(const Point& pos_corr, double cluster_t0, int face, int apa) const override
  {
    return m_sce.filter(pos_corr, cluster_t0, face, apa);
  }

  virtual Dataset forward(const Dataset& pc_in,
                          const std::vector<std::string>& arr_in_names,
                          const std::vector<std::string>& arr_out_names,
                          double cluster_t0, int face, int apa) const override
  {
    auto tmp = m_t0.forward(pc_in, arr_in_names, arr_out_names, cluster_t0, face, apa);
    return m_sce.forward(tmp, arr_out_names, arr_out_names, cluster_t0, face, apa);
  }
-UU-:----F1  PCTransforms.cxx   78% L336  Git:sce-v0.33.0  (C++/l Abbrev) ----------------------------------
```

Build and runtime integration status

- Local WCT install completed successfully.

- Installed libraries include:
  - libWireCellClus.so
  - libWireCellRoot.so

- libWireCellClus.so contains SCECorrection and T0SCECorrection

- larwirecell resolves against the local WCT install, not CVMFS

```
SL7> [sbndgpvm05 11:50:47 AM] wct > ./wcb -j4 -p --notests install
Waf: Entering directory `/exp/sbnd/data/users/abhat/wct_sce/wct/build'
[418/530][78%][/][=========================================================================================>                    ][38m46.595s]
generating doctest main: /exp/sbnd/data/users/abhat/wct_sce/wct/build/aux/wcdoctest-aux.cxx
[419/530][79%][-][==========================================================================================>                   ][38m46.598s]
generating doctest main: /exp/sbnd/data/users/abhat/wct_sce/wct/build/clus/wcdoctest-clus.cxx
[420/530][79%][\][==========================================================================================>                   ][38m46.600s]
generating doctest main: /exp/sbnd/data/users/abhat/wct_sce/wct/build/gen/wcdoctest-gen.cxx
[421/530][79%][|][==========================================================================================>                   ][38m46.601s]
generating doctest main: /exp/sbnd/data/users/abhat/wct_sce/wct/build/iface/wcdoctest-iface.cxx
[422/530][79%][/][===========================================================================================>                  ][38m46.606s]
generating doctest main: /exp/sbnd/data/users/abhat/wct_sce/wct/build/img/wcdoctest-img.cxx
[423/530][79%][-][===========================================================================================>                  ][38m46.607s]
generating doctest main: /exp/sbnd/data/users/abhat/wct_sce/wct/build/pgraph/wcdoctest-pgraph.cxx
[424/530][80%][\][===========================================================================================>                  ][38m46.608s]
generating doctest main: /exp/sbnd/data/users/abhat/wct_sce/wct/build/test/wcdoctest-test.cxx
[425/530][80%][|][============================================================================================>                 ][38m46.610s]
generating doctest main: /exp/sbnd/data/users/abhat/wct_sce/wct/build/util/wcdoctest-util.cxx
[900/900][100%][|][=========================================================================================================================>][51m3.670s]
Waf: Leaving directory `/exp/sbnd/data/users/abhat/wct_sce/wct/build'
'install' finished successfully (51m4.716s)
```

Runtime config status

- What is solved:
  - identified the config hook: switch_scope()
  - confirmed clustering methods already consume pc_transforms

- What is not yet fully solved:
  - end-to-end SBND production config override
  - demonstration that the real runtime chain picks up the local clus.jsonnet
  - physics validation on a full reconstruction job

```
    examine_x_boundary(name="") :: {
        type: "ClusteringExamineXBoundary",
        name: prefix+name,
        data: dv_cfg + scope_cfg,
        uses: [detector_volumes],
    },

    protect_overclustering(name="") :: {
        type: "ClusteringProtectOverclustering",
        name: prefix+name,
        data: dv_cfg + pcts_cfg + scope_cfg,
        uses: [detector_volumes, pc_transforms],
    },

    neutrino(name="", num_try=1) :: {
        type: "ClusteringNeutrino",
        name: prefix+name,
        data: {
            num_try: num_try,
        } + dv_cfg + scope_cfg,
        uses: [detector_volumes],
    },

    switch_scope(name="", correction_name="T0SCECorrection") :: {
        type: "ClusteringSwitchScope",
        name: prefix+name,
        data: {
            correction_name: correction_name,
        } + pcts_cfg + scope_cfg,
        uses: [pc_transforms],
    },

    // This configures RetileCluster, a per-cluster helper for
    // ClusteringRetile as well as others.  Use the sampler() function to
    // provide properly formed elements to the array-of-object argument
    // "samplers".
    retiler(name="", anodes=[], samplers=[], cut_time_low=-1e9, cut_time_high=1e9) :: {
```

Possible adoption path for Wire-Cell SCE

- Can use T0SCECorrection:
  - preserve existing T0 behavior
  - then apply SCE on corrected point clouds
  - minimize disruption to existing clustering logic

$$\text{Raw point cloud} \rightarrow \text{T0Correction} \rightarrow \text{SCECorrection} \rightarrow \text{corrected point cloud}$$

Thoughts on validation plan

- Validation target 1:
  - crossing muons for APA stitching
  - check continuity across boundaries
  - compare stitched geometry before/after SCE

- Validation target 2:
  - cosmic tagger / fiducial volume
  - compare endpoint positions before/after correction
  - quantify effective shrinkage of corrected fiducial region

- Validation target 3:
  - configuration sanity
  - prove local clus.jsonnet override is actually in use
  - run a minimal reconstruction chain with T0SCECorrection

Back Up Slides