Migration of the Silicon tracking algorithm to Gaudi

Yao ZHANG, Chengdong FU, Sheng-Sen SUN
(on behalf of CEPCSW workgroup)

2019 International Workshop on the CepC
Nov 18, Beijing

Contents

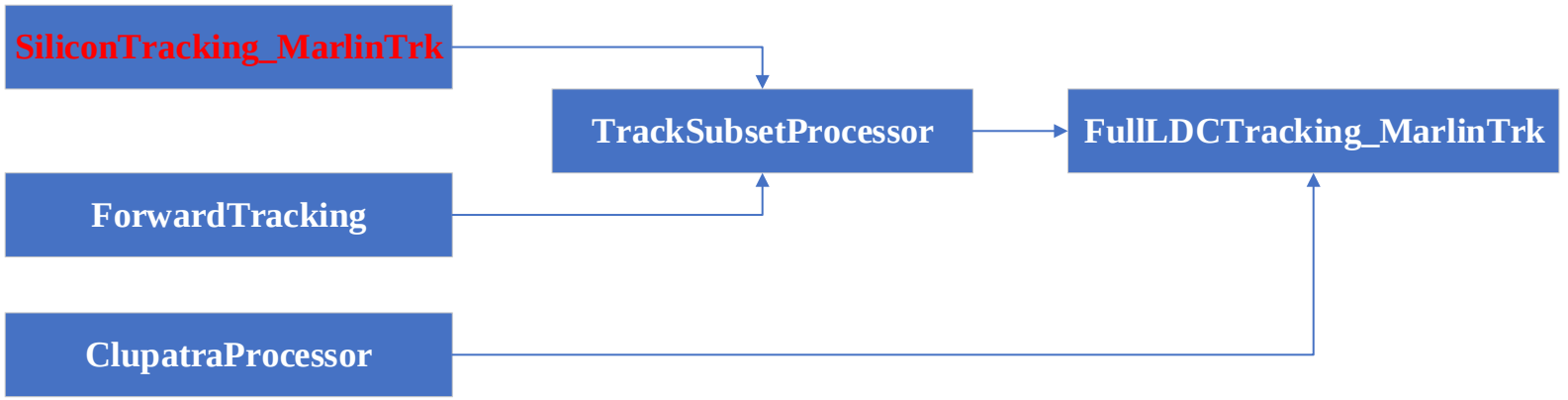
- Introduction
- Realization (How to)
- Status
  - Issues and solutions
- Comparison (usages and results)
- Summary

Introduction

- The Circle Electron Positron Collider is a large international scientific project initiated by and to be hosted in China, one of future particle experiments in plan.
- CepC software serve CepC detector study, requirements toward
  - ~~CDR~~
    - MokkaC & Marlin
  - TDR
    - Reliable
    - Flexible
    - Developable
  - Running
    - Parallel
    - Grid
- A new software framework based on Gaudi (CEPCSW)
- Proposal in Oxford workshop
  - Initial prototype with core software (Jiaheng's) and simulation (Tao's)
  - Demo with simulation and one reconstruction algorithm at less

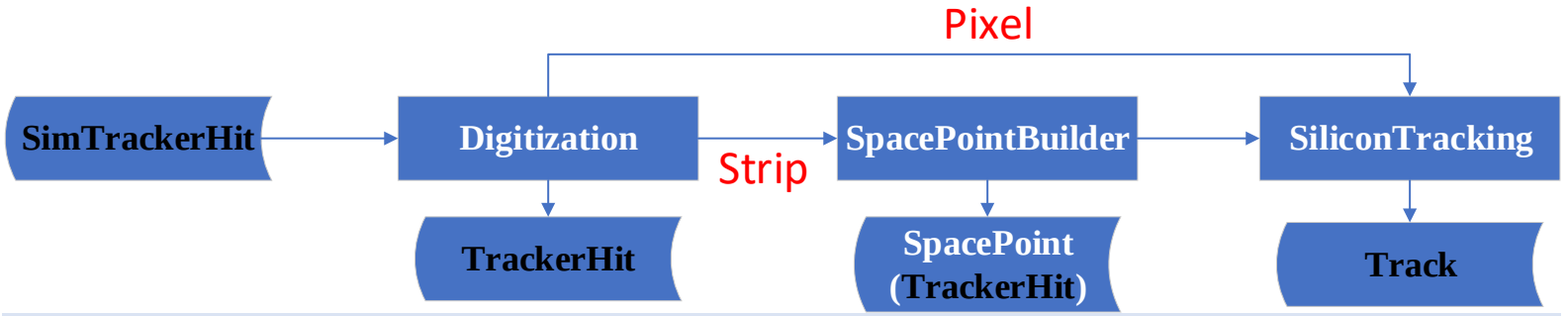
Implement Tracking from Marlin into CEPCSW

- Tracking processes in CDR (Marlin):



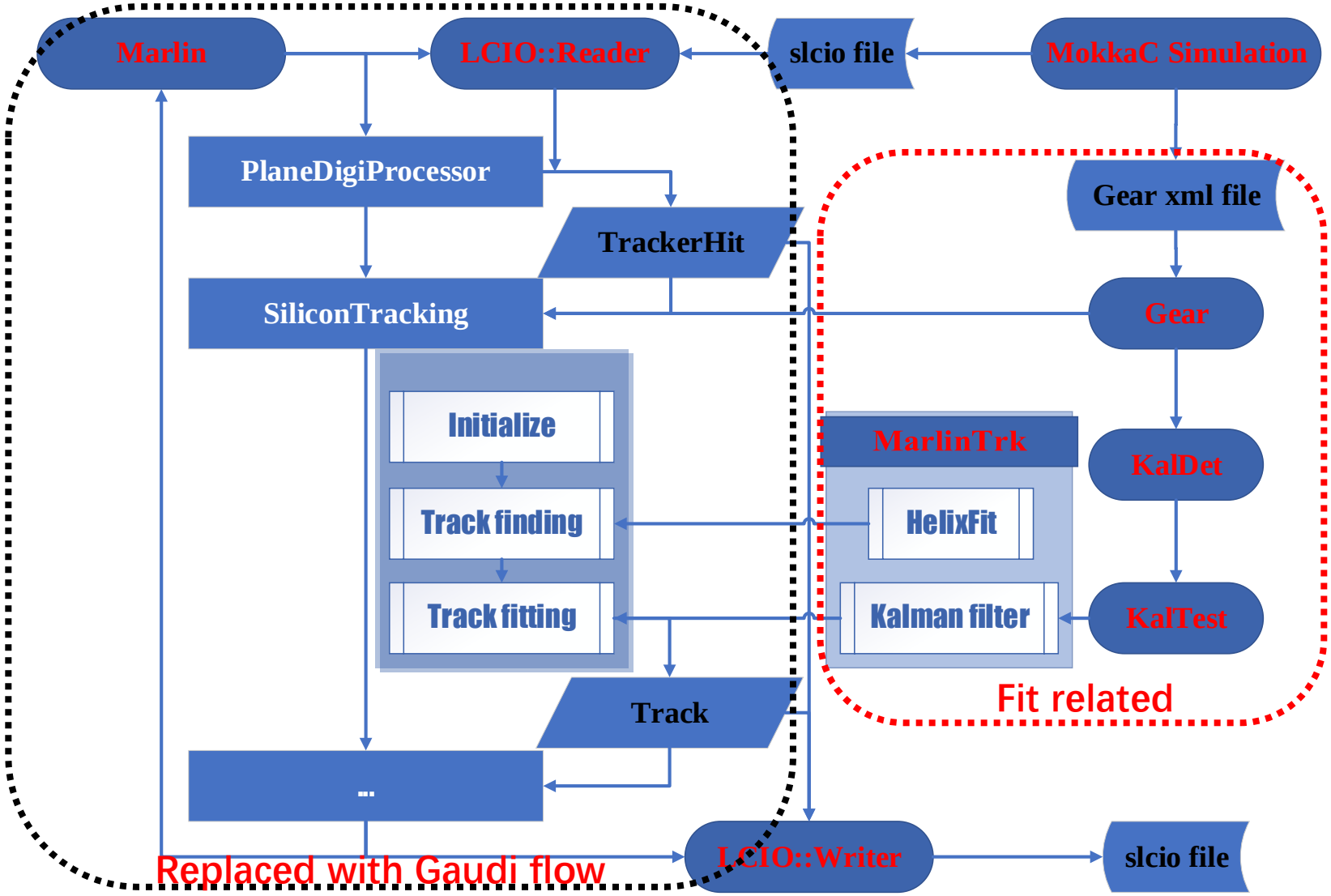
SiliconTracking_MarlinTrk is chosen as the first migrated reconstruction, since it has less dependency and the tracking for silicon detector is more simple than TPC.

- SiliconTracking process



SiliconTracking for vertex detector (pixel VXD) only (without strip SIT) is most simple option.

Working flow



Since CEPCSW's itself Geometry and Track fitting (Kalman filter) services have not bee ready, they are kept as external packages.

Fit related packages (Service)

- Gear (external library)
  - Loaded by a Gaudi service (GearSvc)

```
class IGearSvc: virtual public IService {
  public:
    DeclareInterfaceID(IGearSvc, 0, 1); // major/minor version

    virtual ~IGearSvc() = default;

    // Get the GEAR Manager
    virtual gear::GearMgr* getGearMgr() = 0;

};
```

In Marlin:
static gear::GearMgr* GEAR ;

- MarlinTrk
  - Interface to fitter, loaded by a Gaudi service (TrackSystemSvc)

```
class ITrackSystemSvc: virtual public IService {
 public:
  DeclareInterfaceID(ITrackSystemSvc, 0, 1); // major/minor version

  virtual ~ITrackSystemSvc() = default;

  //Get the track manager
  virtual MarlinTrk::IMarlinTrkSystem* getTrackSystem() = 0;

  virtual void removeTrackSystem() = 0;
};
```

In Marlin:

```
class Factroy{
  public:
    static IMarlinTrkSystem*  createMarlinTrkSystem(const std::string& systemType,
                                                    const gear::GearMgr* mgr ,
                                                    const std::string& options );
```

Fit related packages (Library)

- MarlinTrk
  - HelixFit
  - HelixTrack
  - MarlinKalTest (IMarlinTrkSystem)
  - MarlinKalTestTrack (IMarlinTrack)
  - MarlinTrkUtils ⇒ non-class function: to call Kalman filter fit (KalTest)
    - createFinalisedLCIOTrack
    - createPrefit
    - createFit
    - finaliseLCIOTrack
    - addHitNumbersToTrack
  - ~~DiagnosticsController~~
  - ~~LCIOTrackPropagators~~
  - ~~MarlinTrkDiagnostics~~
- Dependencies
  - LCIO ⇒ plcio
  - GEAR
  - ROOT
  - CLHEP
  - KalTest
  - KalDet
  - streamlog
- CMakeLists.txt

```
class MarlinKalTestTrack : public MarlinTrk::IMarlinTrack {
  public:
    MarlinKalTestTrack(MarlinKalTest* ktest) ;
    ~MarlinKalTestTrack() ;

  private:
    MarlinKalTestTrack(const MarlinKalTestTrack&) ;
    MarlinKalTestTrack& operator=(const MarlinKalTestTrack&);

    int addHit(plcio::TrackerHit* hit) ;
    int addHit(plcio::TrackerHit* trkhit, const ILDVMeasLayer* ml) ;
    ...
```

such as:

int addHit(EVENT::TrackerHit* hit) ;
int addHit(EVENT::TrackerHit* trkhit, const ILDVMeasLayer* ml) ;

```
gaudi_add_library(TrackSystemSvcLib ${TrackSystemSvcLib_srcs}
    PUBLIC_HEADERS TrackSystemSvc
    INCLUDE_DIRS GaudiKernel ROOT CLHEP gear ${LCIO_INCLUDE_DIR
    LINK_LIBRARIES GaudiKernel ROOT CLHEP $ENV{GEAR}/lib/libgea
)
```

Fit related packages (External)

- KalTest: provides base Kalman filter support, dependent on ROOT only
  - geomlib/
  - kallib/
  - kaltracklib/
  - utils/
  - re-compile and link in CEPCSW environment
- KalDet: geometry support package through GEAR
  - gen/
  - ild/
  - kern/
  - lctpc/
  - othertpc/
  - change data model dependency from LCIO to plcio

```
class ILDVMeasLayer : public TVMeasLayer {
  ...
  /** Convert LCIO Tracker Hit to an ILDPLanarTrackHit  */
  virtual ILDVTrackHit* ConvertLCIOTrkHit(plcio::TrackerHit* trkhit) const=0;
  ...
```

virtual ILDVTrackHit* ConvertLCIOTrkHit(EVENT::TrackerHit* trkhit) const = 0 ;

- Migrated as external packages
- streamlog: library in ILCUTIL package
  - does not depend on any external package, therefore does not need re-compile
  - Will be updated to CEPCSW Gaudi print interface

From SiliconTracking_Marlin to SiliconTrackingAlg

- Since **MarlinTrk**, **KalDet** and **KalTest** are all migrated and kept in CEPCSW at first step, less modification needed
- Main algorithm SiliconTrackingAlg

```
class SiliconTracking : public GaudiAlgorithm {
  friend class AlgFactory<SiliconTracking>;
 public:
  SiliconTracking(const std::string& name, ISvcLocator* svcLoc);
  virtual StatusCode initialize() ;
  virtual StatusCode execute() ;
  virtual StatusCode finalize() ;

 protected:
  ...
  MarlinTrk::HelixFit* _fastfitter;
  gear::GearMgr* _GEAR;
  MarlinTrk::IMarlinTrkSystem* _trksystem ;

  Gaudi::Property<bool> _MSOn{this, "MultipleScatteringOn",  true};
  Gaudi::Property<bool> _ElossOn{this, "EnergyLossOn",       true};
  Gaudi::Property<bool> _SmoothOn{this, "SmoothOn",          true};
  ...
  DataHandle<plcio::EventHeaderCollection> _headerCol          {"EventHeaderCol", Gaudi::DataHandle::Reader, this};
  DataHandle<plcio::TrackerHitCollection>  _inVTXColHdl        {"VXDTrackerHits", Gaudi::DataHandle::Reader, this};
  DataHandle<plcio::TrackerHitCollection> _inFTDPixelColHdl{"FTDPixelTrackerHits", Gaudi::DataHandle::Reader, this};
  DataHandle<plcio::TrackerHitCollection> _inFTDSpacePointColHdl{"FTDSpacePoints", Gaudi::DataHandle::Reader, this};
  DataHandle<plcio::TrackerHitCollection> _inSITColHdl         {"SITTrackerHits", Gaudi::DataHandle::Reader, this};
  DataHandle<plcio::TrackCollection>      _outColHdl                 {"SiTracks", Gaudi::DataHandle::Writer, this};
  DataHandle<plcio::LCRelationCollection> _outRelColHdl   {"TrackerHitRelations", Gaudi::DataHandle::Writer, this};
  ...
```

Values and names input

parameters input
data input and output

- Support classes (CaloHitExtended ClusterExtended GroupTracks HelixClass LineClass TrackerHitExtended TrackExtended) from **MarlinUtil**
  - Change data model from LCIO to plcio, just like **MarlinTrk** and **KalDet**

Difference of Usage with Marlin Processor

```
SiliconTracking_MarlinTrk::SiliconTracking_MarlinTrk() : Processor("SiliconTracking_MarlinTrk") {
  ...
  registerProcessorParameter("MultipleScatteringOn", "Use MultipleScattering in Fit", _MSOn, bool(true));
  ...
}

void SiliconTracking_MarlinTrk::init() {
  ...
  _trksystem =  MarlinTrk::Factory::createMarlinTrkSystem( "KalTest" , marlin::Global::GEAR , "" );
  if( _trksystem == 0 ){
    throw EVENT::Exception( std::string("Cannot initialize MarlinTrkSystem of Type: KalTest" ) );
  }
  _trksystem->setOption( IMarlinTrkSystem::CFG::useQMS,       _MSOn ) ;
  _trksystem->setOption( IMarlinTrkSystem::CFG::usedEdx,      _ElossOn) ;
  _trksystem->setOption( IMarlinTrkSystem::CFG::useSmoothing,  _SmoothOn) ;
  _trksystem->init() ;
  ...
}

void SiliconTracking_MarlinTrk::processEvent( LCEvent * evt ) {
  ...
  try {
    LCCollection * hitCollection = evt->getCollection("VXDTrackerHits");
    int nelem = hitCollection->getNumberOfElements();
    for (int ielem=0; ielem<nelem; ++ielem) {
      TrackerHitPlane* hit = dynamic_cast<EVENT::TrackerHitPlane*>(hitCollection->getElementAt(ielem));
      ...
    }
  }
  catch(DataNotAvailableException &e){
    ...
  }
  ...
  bool backwards = IMarlinTrack::backward;
  MarlinTrk::IMarlinTrack* marlinTrk = _trksystem->createTrack();
  int error = 0;
  try {
    error = MarlinTrk::createFinalisedLCIOTrack(marlinTrk, trkHits, track, backwards, cov, _bField, _maxChi2);
  }
  catch (...) {
  ...
  }
  ...
}
```

```
SiliconTracking::SiliconTracking(const std::string& name, ISvcLocator* svcLoc) : GaudiAlgorithm(name, svcLoc) {
  ...
  declareProperty("HeaderCol", _headerCol);
  declareProperty("VTXHitCollection", _inVTXColHdl, "Handle of the Input VTX TrackerHits collection");
  ...
}

StatusCode  SiliconTracking::initialize() {
  ...
  auto _trackSystemSvc = service<ITrackSystemSvc> ("TrackSystemSvc");
  if ( !_trackSystemSvc ) {
    error() << "Failed to find TrackSystemSvc ..." << endmsg;
    return StatusCode::FAILURE;
  }
  _trksystem =  _trackSystemSvc->getTrackSystem();
  if( _trksystem == 0 ){
    error() << "Cannot initialize MarlinTrkSystem of Type: KalTest" <<endmsg;
    return StatusCode::FAILURE;
  }
  _trksystem->setOption( IMarlinTrkSystem::CFG::useQMS,       _MSOn ) ;
  _trksystem->setOption( IMarlinTrkSystem::CFG::usedEdx,      _ElossOn) ;
  _trksystem->setOption( IMarlinTrkSystem::CFG::useSmoothing,  _SmoothOn) ;
  _trksystem->init() ;
  ...
}

StatusCode SiliconTracking::execute(){
  ...
  const plcio::TrackerHitCollection* hitVTXCol = nullptr;
  try {
    hitVTXCol = _inVTXColHdl.get();
  }
  catch ( GaudiException &e ) {
    debug() << "Collection " << _inVTXColHdl.fullKey() << " is unavailable in event " << _nEvt << endmsg;
    success = 0;
  }
  if(hitVTXCol){
    int nelem = hitVTXCol->size();
    for (int ielem=0; ielem<nelem; ++ielem) {
      plcio::TrackerHit hit = hitVTXCol->at(ielem);
      ...
    }
  }
  ...
  bool fit_backwards = IMarlinTrack::backward;
  MarlinTrk::IMarlinTrack* marlinTrk = _trksystem->createTrack();
  int status = 0;
  try {
    status = MarlinTrk::createFinalisedLCIOTrack(marlinTrk, trkHits, &track, fit_backwards, covMatrix, _bField, _maxChi2PerHit);
  } catch (...) {...}
  ...
}
```

Interface load

Issues and Solutions (Temporary)

- Can't get relative object directly in plcio data model
  - Remove relative object usage in tracking and after fitting, influence some following analysis based on MC truth particles
  - Going to extend plcio with a service to handle the object relations
- Can't convert pointers of data objects each other
  - Fix data type in algorithm and use template in codes of module classes
  - Only use one type tracker hit as temporary now, saving U,V directions and resolutions into CovMatrix

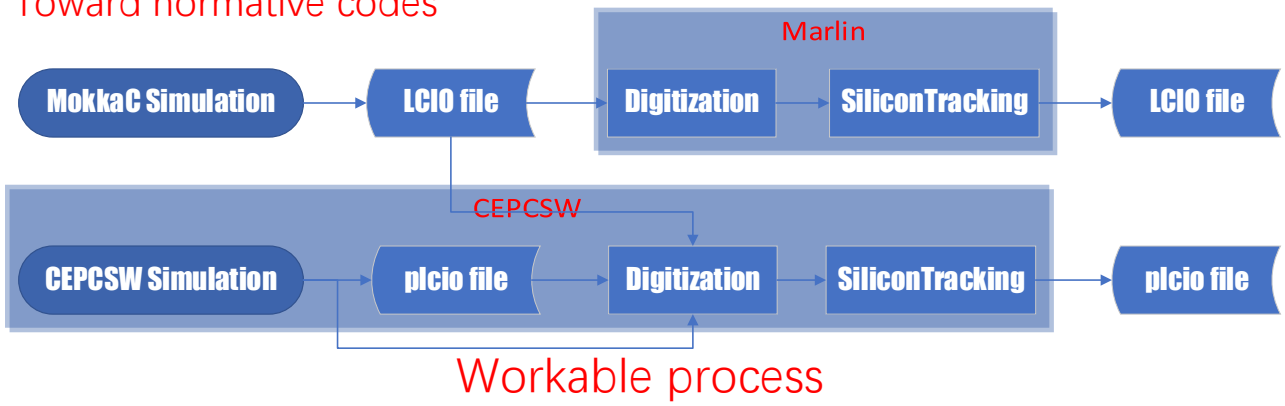
```
auto trkHit = trkhitVec->create();
std::array<float, 6> cov;
cov[0] = u_direction[0];
cov[1] = u_direction[1];
cov[2] = resU;
cov[3] = v_direction[0];
cov[4] = v_direction[1];
cov[5] = resV;
trkHit->setCovMatrix(cov);
```

```
EVENT::TrackerHitPlaneImpl* trkHit = new EVENT::TrackerHitPlaneImpl ;
trkHit->setU( u_direction ) ;
trkHit->setV( v_direction ) ;
trkHit->setdU( resU ) ;
trkHit->setdV( resV ) ;
```

- From TrackerHit pointer (LCIO) to TrackerHit object (plcio)
  - Plcio's hit object does not have data member, just only data object pointer
  - In order to keep pointer usage in KalDet and TrackSystemSvc, build temporary object vector for each event
    - _allHits.reserve(100000);
    - _allHits.push_back(hit); // &hit can be used as plcio::TrackerHit* before clear()
    - _allHits.clear();
  - Will be updated to object transmition
- Geometry service during the transition from Marlin to Gaudi
  - Keep the GEAR package in Marlin at present, similar to load as TrackSystemSvc
- keep some nonstandard use for Gaudi
  - _fastfitter = new MarlinTrk::HelixFit();
- Keep LCIO's UTIL dependency for cell id encoder

Status

- CEPCSW framework ready! (Jiaheng's)
- Simulation tools in CEPCSW ready! (Tao's)
- One reconstruction algorithm — SiliconTrackingAlg
  - Dependencies
    - TrackSystemSvc (service): ready
    - TrackSystemSvcLib (library): ready
    - KalDet (external): ready
    - KalTest (exrernal): ready
    - GEAR (service): ready
    - Data input (slcio file convert to plcio): ready
    - Data conversion (digitization): ready
  - Algorithm
    - Temporary solution scheme: ready
    - Toward normative codes



Usage Changes

- From xml file options to python options

```
#!/usr/bin/env python
from Gaudi.Configuration import *
from Configurables import LCIODataSvc, CEPCDataSvc
...
from Configurables import TrackSystemSvc
tracksystemsvc = TrackSystemSvc("TrackSystemSvc")
vxdhitname = "VXDTrackerHits"

from Configurables import PlanarDigiAlg
digi = PlanarDigiAlg("PlanarDigiAlg")
digi.SimTrackHitCollection = "VXDCollection"
digi.TrackerHitCollection = vxdhitname
digi.ResolutionU = [0.0028, 0.006, 0.004, 0.004, 0.004, 0.004]
digi.ResolutionV = [0.0028, 0.006, 0.004, 0.004, 0.004, 0.004]

from Configurables import SiliconTracking
tracking = SiliconTracking("SiliconTracking")
tracking.VTXHitCollection = vxdhitname
tracking.UseSIT = 0
tracking.LayerCombinations = [5,3,1, 5,3,0, 5,2,1, 5,2,0, 4,3,1, 4,3,0, 4,2,1,
4,2,0]
tracking.SmoothOn = 0

from Configurables import PodioOutput
plcioout = PodioOutput("PlcioWriter")
plcioout.filename = "reconstruction.root"
plcioout.outputCommands = ["keep *"]

# ApplicationMgr
from Configurables import ApplicationMgr
ApplicationMgr( TopAlg = [lcioinput, plcioread, digi, tracking, plcioout],
                EvtSel = 'NONE',
                EvtMax = 100000,
                ExtSvc = [rsvc, wsvc, evtseeder, gearsvc, tracksystemsvc]
)
```

```
<marlin>
  <execute>
    <processor name="MyAIDAProcessor"/>
    <processor name="VXDPlanarDigiProcessor"/>
    <processor name="MySiliconTracking_MarlinTrk"/>
    ...
  </execute>
  <global>
    <parameter name="LCIOInputFiles">
      simulation.slcio
    </parameter>
    ...
  </global>
  ...
  <processor name="VXDPlanarDigiProcessor" type="PlanarDigi
    <parameter name="ResolutionU" type="float">
      0.0028 0.006 0.004 0.004 0.004 0.004
    </parameter>
    <parameter name="ResolutionV" type="float">
      0.0028 0.006 0.004 0.004 0.004 0.004
    </parameter>
    <parameter name="SimTrackHitCollectionName" type="strin
    <parameter name="TrackerHitCollectionName" type="string
  </processor>

  <processor name="MySiliconTracking_MarlinTrk" type="Silic
    <parameter name="VTXHitCollectionName" type="string" lo
    <parameter name="UseSIT" type="int">0 </parameter>
    <parameter name="LayerCombinations" type="IntVec">
      5 3 1  5 3 0  5 2 1  5 2 0
      4 3 1  4 3 0  4 2 1  4 2 0
    </parameter>
    <parameter name="SmoothOn" type="bool"> false </paramet
  </processor>
  ...
</marlin>
```

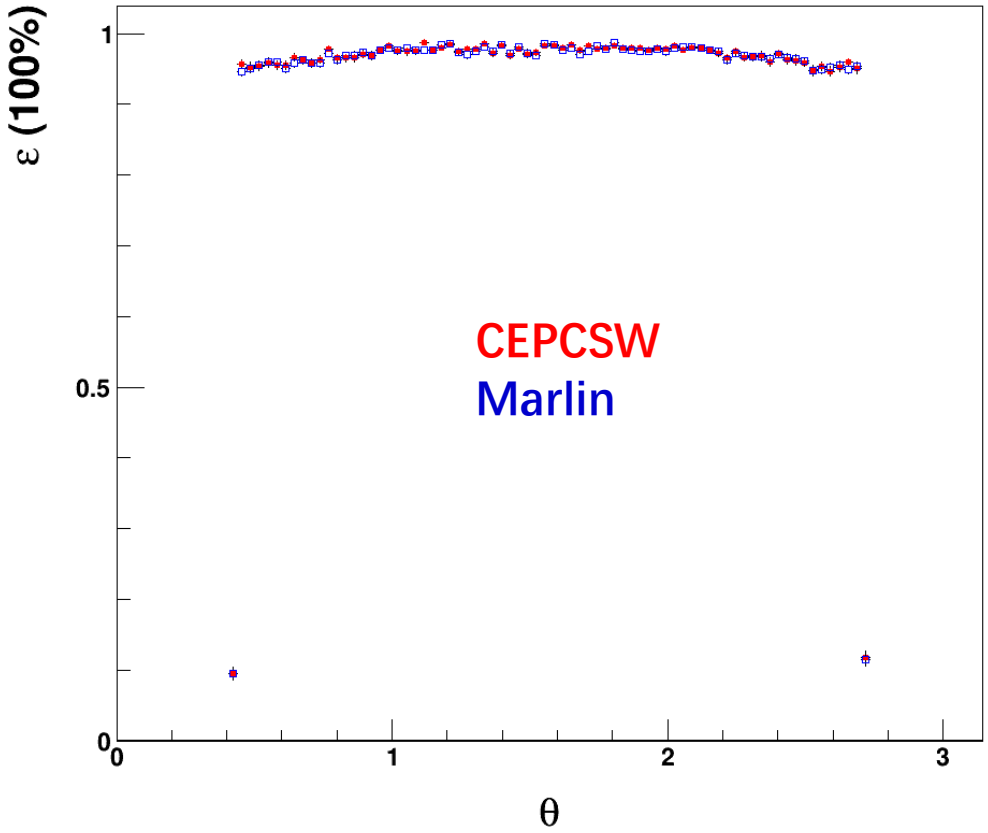
Comparison of tracking and fitting

- Single muon particle (p=10 GeV/c, $\theta \in [10°,170°]$, $\phi \in [0°,360°]$)
  - same input (simulated by MokkaC with only VXD)
  - same reconstruction options

Tracking efficiency

- Definition
  - $\varepsilon = N_{macthed\_track}/N_{MC(primary)}$
  - Matching:
    - $|par_{fit}-par_{MC}|<5\sigma_{par}$ (par=d0, phi0, ω, z0, tanλ)

	R (mm)	$\lvert z\rvert$ (mm)	$\lvert\cos\theta\rvert$	σ(μm)
Layer 1	16	62.5	0.97	2.8
Layer 2	18	62.5	0.96	6
Layer 3	37	125.0	0.96	4
Layer 4	39	125.0	0.95	4
Layer 5	58	125.0	0.91	4
Layer 6	60	125.0	0.90	4



Fake rate:

CEPCSW: (1.23±0.04)%

Marlin: (1.21±0.04)%

⇓

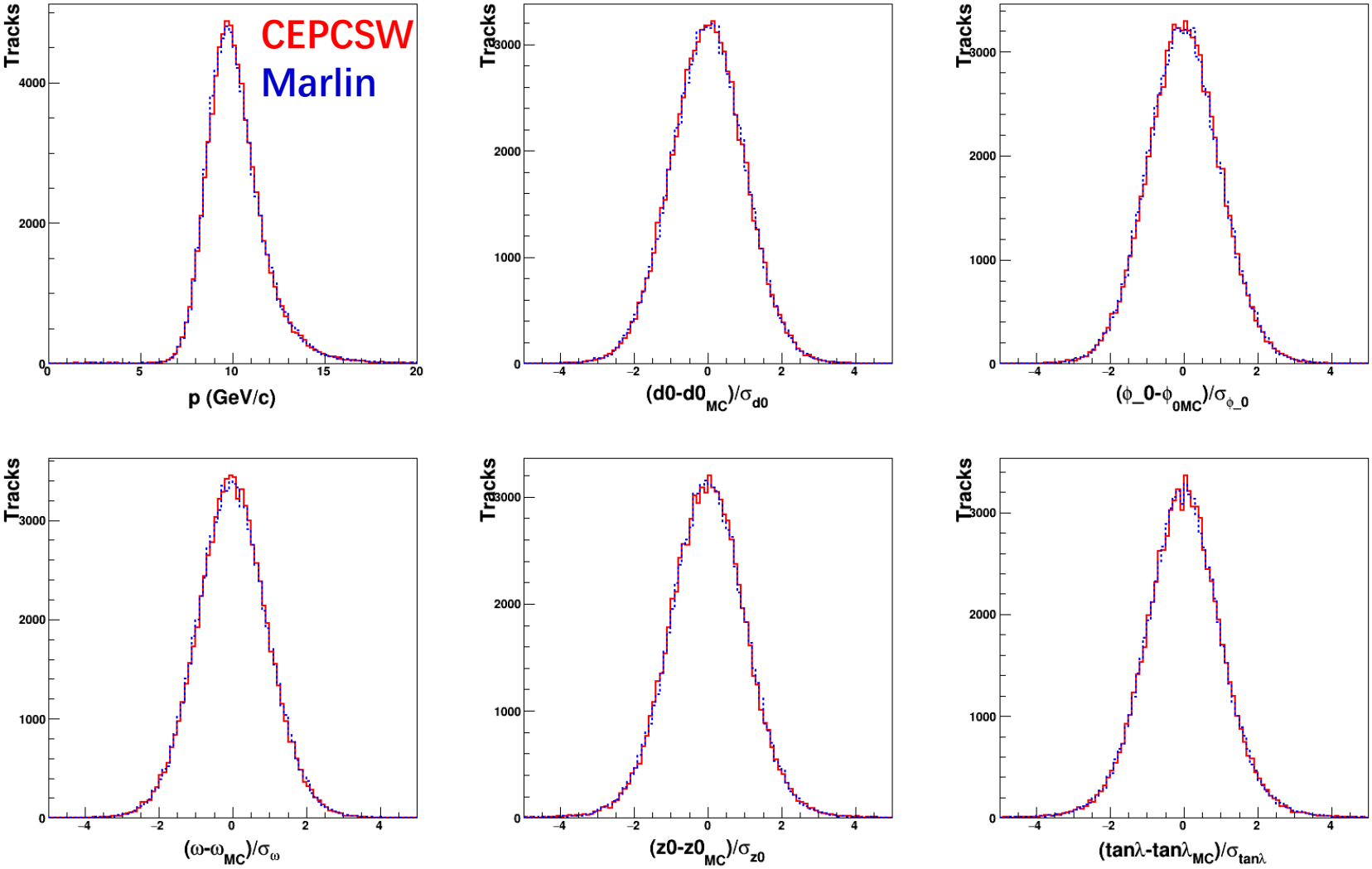
10σ cut:

CEPCSW: (0.56±0.03)%

Marlin: (0.56±0.03)%

Resolution & Pull

- Only three doubly-layer pixel used →



CEPCSW's results are consistent with Marlin's, but they are not identical. Why?

Comparison of tracking and fitting

- Single muon particle (p=10 GeV/c, $\theta \in [10°,170°]$, $\phi \in [0°,360°]$)
  - same input (simulated by MokkaC with only VXD)
  - same reconstruction options

- The difference is caused by different random number in digitization (gaussian smear)
  - if testing by set space resolution as 0, and debugging, they are fully identical
  - CEPCSW

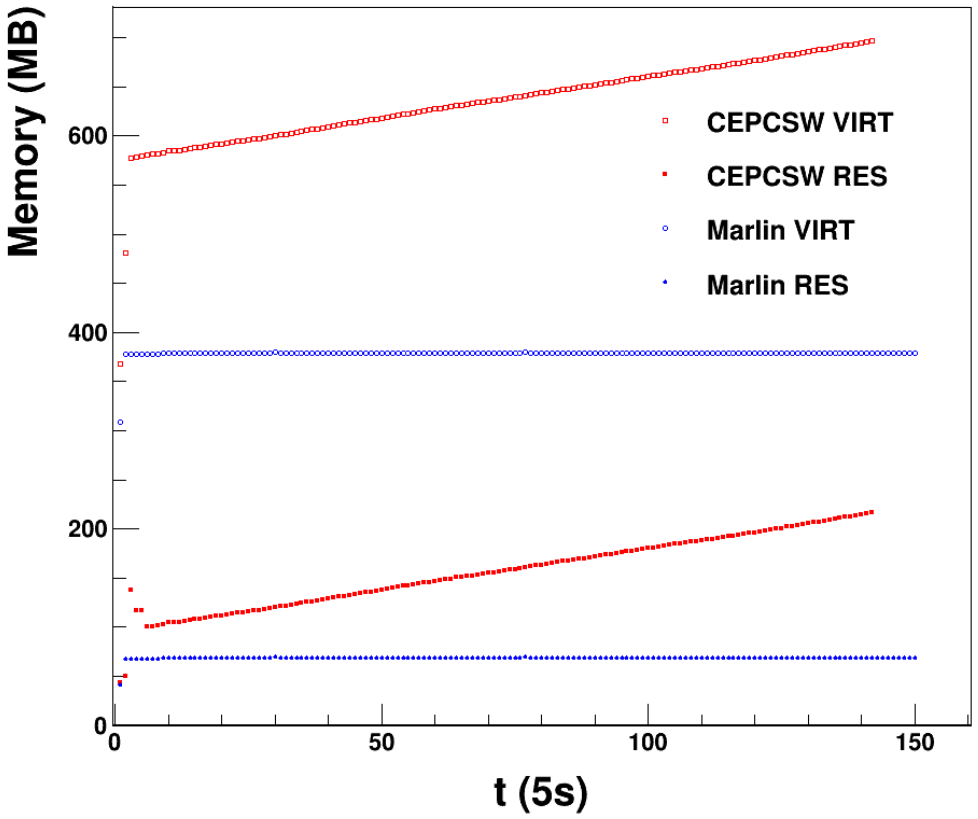
```
SiliconTracking     DEBUG Total 4-momentum of Si Tracks : E = 375.662 Px = -0.703028 Py = -375.661 Pz = -0.0104615
SiliconTracking     DEBUG Total 4-momentum of Si Tracks : E = 453.497 Px = 196.305 Py = 408.808 Pz = -0.0370978
SiliconTracking     DEBUG Total 4-momentum of Si Tracks : E = 1205.55 Px = 661.275 Py = -1008 Pz = 0.128028
SiliconTracking     DEBUG Total 4-momentum of Si Tracks : E = 488.419 Px = 486.218 Py = 46.3153 Pz = -0.0540399
SiliconTracking     DEBUG Total 4-momentum of Si Tracks : E = 2622.19 Px = -1277.83 Py = -2289.76 Pz = -0.177663
SiliconTracking     DEBUG Total 4-momentum of Si Tracks : E = 2642.46 Px = 2284.13 Py = 1328.66 Pz = 0.146478
SiliconTracking     DEBUG Total 4-momentum of Si Tracks : E = 656.724 Px = 18.8581 Py = -656.453 Pz = -0.0383404
SiliconTracking     DEBUG Total 4-momentum of Si Tracks : E = 512.21 Px = 119.476 Py = -498.081 Pz = -0.0241696
SiliconTracking     DEBUG Total 4-momentum of Si Tracks : E = 2563.15 Px = -1801.14 Py = -1823.63 Pz = -0.0922066
SiliconTracking     DEBUG Total 4-momentum of Si Tracks : E = 405.058 Px = -380.559 Py = -138.733 Pz = 0.018916
```

  - Marlin

```
[ DEBUG4 "MySiliconTracking_MarlinTrk"] Total 4-momentum of Si tracks : E = 3.756618e+02 Px = -7.335976e-01 Py = -3.756611e+02 Pz = -1.046148e-02
[ DEBUG4 "MySiliconTracking_MarlinTrk"] Total 4-momentum of Si tracks : E = 4.534971e+02 Px = 1.963345e+02 Py = 4.087937e+02 Pz = -3.709782e-02
[ DEBUG4 "MySiliconTracking_MarlinTrk"] Total 4-momentum of Si tracks : E = 1.205546e+03 Px = 6.612491e+02 Py = -1.008014e+03 Pz = 1.280278e-01
[ DEBUG4 "MySiliconTracking_MarlinTrk"] Total 4-momentum of Si tracks : E = 4.884193e+02 Px = 4.862154e+02 Py = 4.634652e+01 Pz = -5.403987e-02
[ DEBUG4 "MySiliconTracking_MarlinTrk"] Total 4-momentum of Si tracks : E = 2.622189e+03 Px = -1.277834e+03 Py = -2.289763e+03 Pz = -1.776625e-01
[ DEBUG4 "MySiliconTracking_MarlinTrk"] Total 4-momentum of Si tracks : E = 2.642462e+03 Px = 2.284150e+03 Py = 1.328632e+03 Pz = 1.464776e-01
[ DEBUG4 "MySiliconTracking_MarlinTrk"] Total 4-momentum of Si tracks : E = 6.567236e+02 Px = 1.884191e+01 Py = -6.564533e+02 Pz = -3.834036e-02
[ DEBUG4 "MySiliconTracking_MarlinTrk"] Total 4-momentum of Si tracks : E = 5.122102e+02 Px = 1.194612e+02 Py = -4.980847e+02 Pz = -2.416962e-02
[ DEBUG4 "MySiliconTracking_MarlinTrk"] Total 4-momentum of Si tracks : E = 2.563147e+03 Px = -1.801128e+03 Py = -1.823639e+03 Pz = -9.220656e-02
[ DEBUG4 "MySiliconTracking_MarlinTrk"] Total 4-momentum of Si tracks : E = 4.050581e+02 Px = -3.805486e+02 Py = -1.387617e+02 Pz = 1.891596e-02
```

Computing Resource Occupy

- Test with 100 000 muons
  - Close CPU time
  - CEPCSW:
    - Memory leak need to fix



CPU:

CEPCSW:

User time (s): 706.94

System time (s): 10.38

Marlin:

User time (s): 755.65

System time (s): 4.03

Manpower of tracking migration

- CEPCSW framework: Jiaheng ZOU
- CEPCSW simulation: Tao LIN
- Migration
  - MarlinTrk, KalDet, KalTest: Chengdong FU
  - PlaneDigiProcessor: Jiaheng ZOU & Fenfen AN
  - SiliconTracking: Chengdong FU

  - SpacePointBuilder: Yao ZHANG
  - FowardTracking: opened
  - ClupatraProcessor: opened
  - TrackSubetProcessor: opened
  - FullLDCTracking: opened

- Detail validation

Summary

- In CEPCSW framework, a reconstruction algorithm SiliconTracking has been migrated from Marlin, and workable
  - In order to simply job, some temporary methods are used, and will be updated in future
  - Most of time while Migrating on interface and data model change
- Results by CEPCSW and Marlin are intelligible same
- Some issues such as memory leak need to solve
- **Todo**
  - Update codes to normative
    - If memory leak still, to fix
  - Create ourselves' cell id encoder in future?
  - Replace GEAR with DD4hep Geometry support?
  - Migrate more reconstruction packages
  - Detail validation

Thanks!