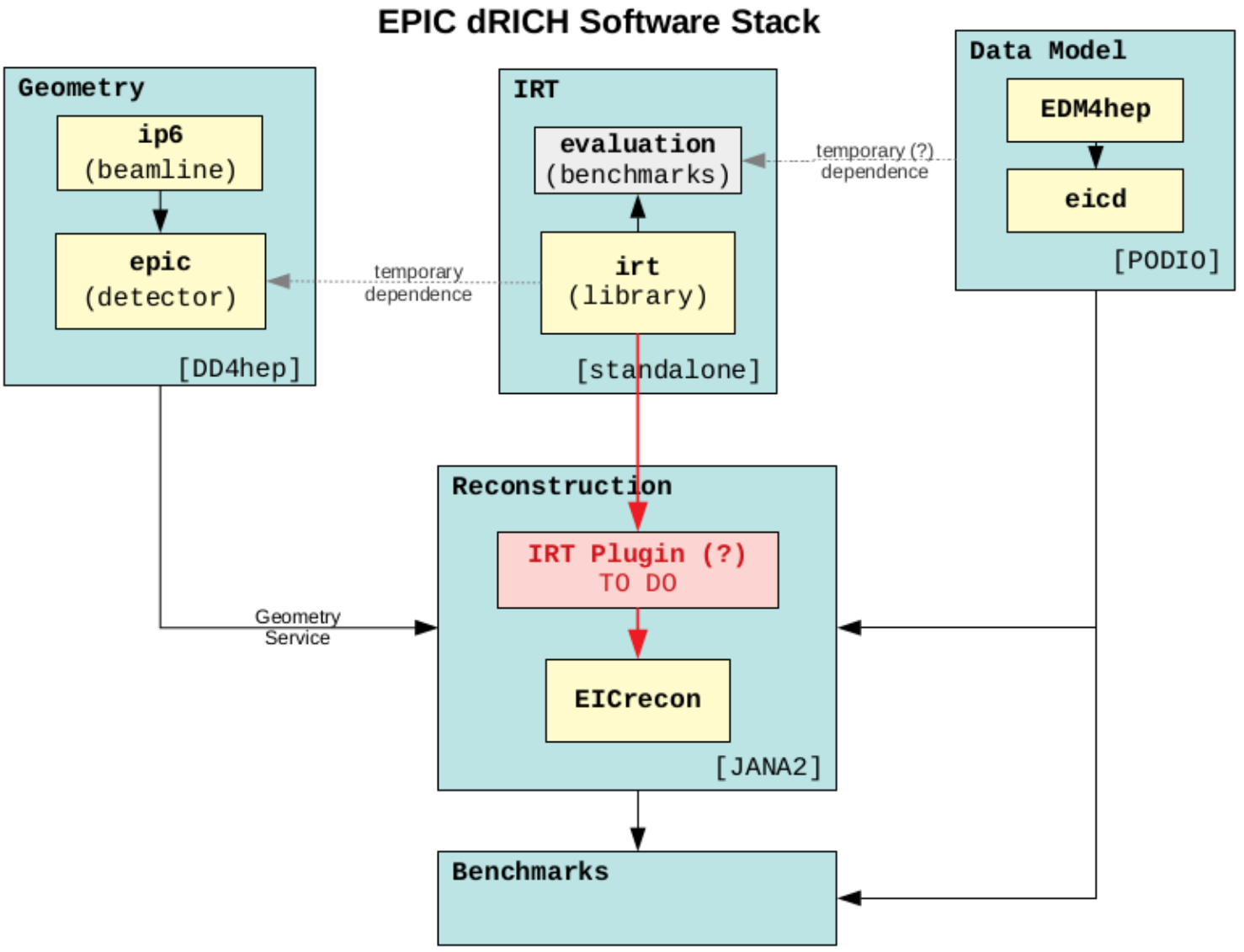
EPIC dRICH Software Update

- EPIC Software Stack
- IRT Integration Status
- Automated Parameter Variation
- Next Steps

Christopher Dilks

dRICH Meeting

3 August 2022

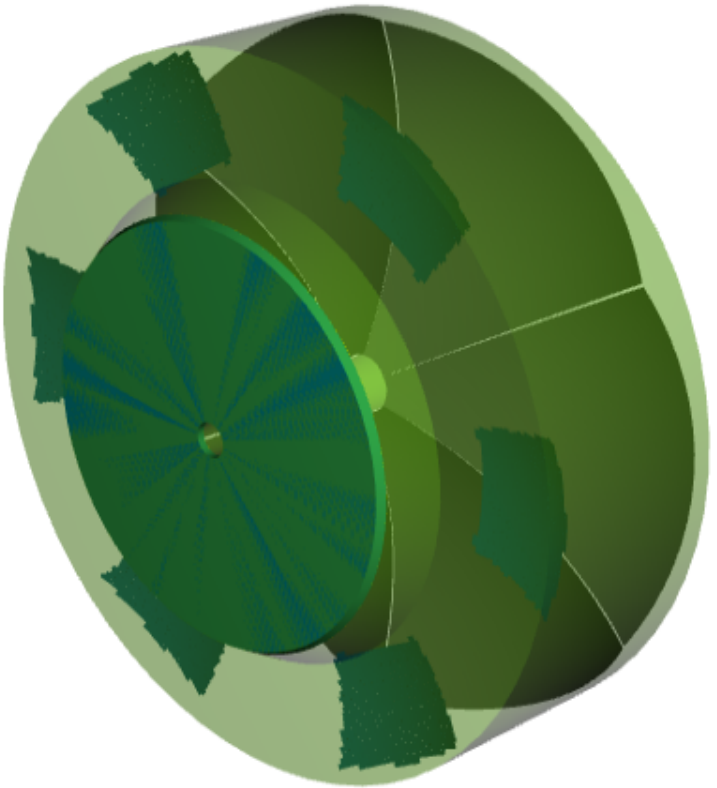

EPIC dRICH Software Stack
Geometry
ip6
(beamline)
epic
(detector)
[DD4hep]
IRT
evaluation
(benchmarks)
irt
(library)
[standalone]
temporary
dependence
temporary (?)
dependence
Data Model
EDM4hep
eicd
[PODIO]
Reconstruction
IRT Plugin (?)
TO DO
EICrecon
[JANA2]
Geometry
Service
Benchmarks

DD4hep Geometry: epic

https://github.com/eic/epic

- dRICH Source file and Compact file
  - Numbers are in an XML file → can be changed ‘on-the-fly’
  - TGeo objects constructed and placed by C++ source code
  - DD4hep : TGeo → Geant4 → simulation
- Material property tables (XML)
  - (TODO: use the common optics header)
- Contains all other EPIC subsystem descriptions

```
<mirror
  material="Acrylic_DRICH"
  surface="MirrorSurface_DRICH"
  vis="DRICH_mirror_vis"
  backplane="DRICH_window_thickness + 0.71*cm"
  rmin="DRICH_rmin1 + DRICH_wall_thickness - 1.0*cm"
  rmax="DRICH_rmax2 - DRICH_wall_thickness - 3.0*cm"
  phiw="59.5*degree"
  thickness="0.2*cm"
  focus_tune_x="69.78*cm"
  focus_tune_z="51.45*cm"
  />
```



Reconstruction Framework

https://github.com/eic/EICrecon

Data Model

MCParticles

Truth particles

- for standalone dRICH studies, these can be our “tracks”
- for a real study, need to use tracker branches

```
edm4hep::MCParticle:
  Description: "The Monte Carlo particle - based on the lcio::MCParticle."
  Author : "F.Gaede, DESY"
  Members:
    - int32_t PDG                          //PDG code of the particle
    - int32_t generatorStatus              //status of the particle as defined by the generator
    - int32_t simulatorStatus              //status of the particle from the simulation program - use BIT constants below
    - float charge                        //particle charge
    - float time                          //creation time of the particle in [ns] wrt. the event, e.g. for preassigned decays or deca
    - double mass                         //mass of the particle in [GeV]
    - edm4hep::Vector3d vertex              //production vertex of the particle in [mm].
    - edm4hep::Vector3d endpoint            //endpoint of the particle in [mm]
    - edm4hep::Vector3f momentum             //particle 3-momentum at the production vertex in [GeV]
    - edm4hep::Vector3f momentumAtEndpoint   //particle 3-momentum at the endpoint in [GeV]
    - edm4hep::Vector3f spin                 //spin (helicity) vector of the particle.
    - edm4hep::Vector2i colorFlow                //color flow as defined by the generator
  OneToManyRelations:
    - edm4hep::MCParticle parents // The parents of this particle.
    - edm4hep::MCParticle daughters // The daughters this particle.
  MutableExtraCode:
    includes: "#include <cmath>"
    declaration: "
    int32_t  set_bit(int32_t val, int num, bool bitval){ return (val & ~(1<<num)) | (bitval << num); }
    void setCreatedInSimulation(bool bitval) { setSimulatorStatus( set_bit( getSimulatorStatus() , BITCreatedInSimulation , bitva
    void setBackscatter(bool bitval) { setSimulatorStatus( set_bit( getSimulatorStatus() , BITBackscatter , bitval ) ) ;   }
    void setVertexIsNotEndpointOfParent(bool bitval) { setSimulatorStatus( set_bit( getSimulatorStatus() , BITVertexIsNotEndpoint
    void setDecayedInTracker(bool bitval) { setSimulatorStatus( set_bit( getSimulatorStatus() , BITDecayedInTracker , bitval ) )
```

…

Data Model

DRICHHits

```
#----------- SimTrackerHit
edm4hep::SimTrackerHit:
  Description: "Simulated tracker hit"
  Author : "F.Gaede, DESY"
  Members:
    - uint64_t cellID      //ID of the sensor that created this hit
    - float EDep                    //energy deposited in the hit [GeV].
    - float time                    //proper time of the hit in the lab frame in [ns].
    - float pathLength              //path length of the particle in the sensitive material that resulted in this hit.
    - int32_t   quality                 //quality bit flag.
    - edm4hep::Vector3d position    //the hit position in [mm].
    - edm4hep::Vector3f momentum    //the 3-momentum of the particle at the hits position in [GeV]
  OneToOneRelations:
    - edm4hep::MCParticle MCParticle //MCParticle that caused the hit.
  MutableExtraCode:
    includes: "#include <cmath>"
    declaration: "
    int32_t  set_bit(int32_t val, int num, bool bitval){ return (val & ~(1<<num)) | (bitval << num); }\n
    void setOverlay(bool val) { setQuality( set_bit( getQuality() , BITOverlay , val ) ) ;   }\n
    void setProducedBySecondary(bool val) { setQuality( set_bit( getQuality() , BITProducedBySecondary , val ) ) ;   }\n
    "
  ExtraCode:
    declaration: "
    static const int  BITOverlay = 31;\n
    static const int  BITProducedBySecondary = 30;\n
    bool isOverlay() const { return getQuality() & (1 << BITOverlay) ; }\n
    bool isProducedBySecondary() const { return getQuality() & (1 << BITProducedBySecondary) ; }\n
    double x() const {return getPosition()[0];}\n
    double y() const {return getPosition()[1];}\n
    double z() const {return getPosition()[2];}\n
    double rho() const {return sqrt(x()*x() + y()*y());}\n
    "
```

Data Model: Updates for IRT

MR: https://eicweb.phy.anl.gov/EIC/eicd/-/merge_requests/70

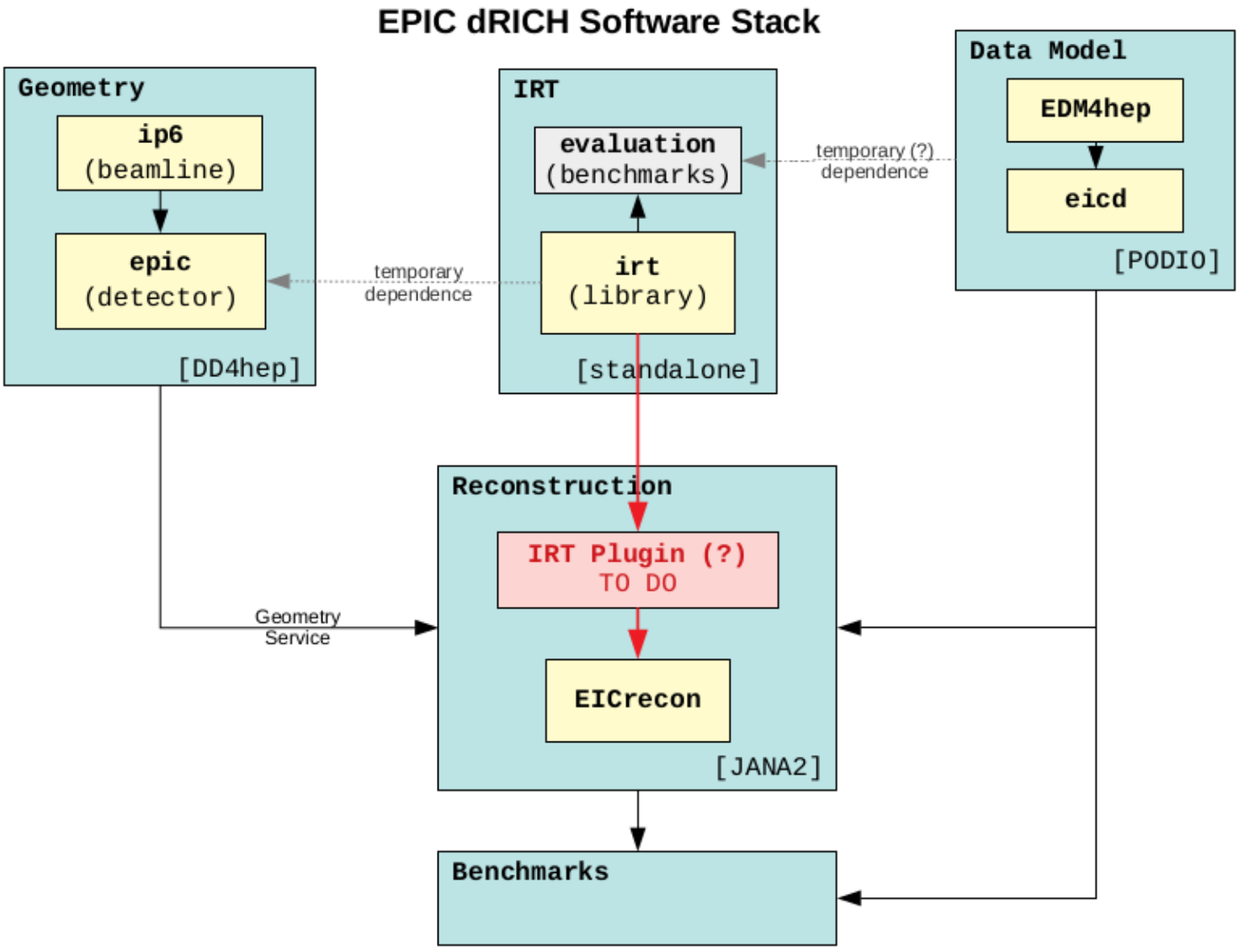
new components:

```
## PID hypothesis from Cherenkov detectors
eicd::CherenkovPdgHypothesis:
  Members:
    - char              radiator          // Radiator number (0/1/..) in a sequence of the IRTAlgorithm configuration file
    - int32_t           pdg               // PDG code
    - float             npe               // Overall p.e. count associated with this hypothesis for a given track
    - float             weight            // The weight associated with this hypothesis (the higher the more probable)

## Cherenkov angle measurement for a given radiator
eicd::CherenkovThetaAngleMeasurement:
  Members:
    - char              radiator          // Radiator number (0/1/..) in a sequence of the IRTAlgorithm configuration file
    - float             npe               // Overall p.e. count associated with this estimate
    - float             theta             // Cherenkov theta angle
    - float             rindex            // Average refractive index for this collection of photons
    - float             wavelength        // Average wavelength for this collection of photons
```

new datatypes:

```
eicd::CherenkovParticleID:
  Description: "Cherenkov detector PID"
  Author: "A. Kiselev, C. Dilks"
  VectorMembers:
    - eicd::CherenkovPdgHypothesis options        // Evaluated PDG hypotheses, typically (e/pi/K/p)
    - eicd::CherenkovThetaAngleMeasurement angles // Evaluated Cherenkov angles for different radiators
  OneToOneRelations:
    ## @TODO: should it be one-to-one?
    - eicd::ReconstructedParticle associatedParticle // associated reconstructed particle
```


EPIC dRICH Software Stack
Geometry
ip6
(beamline)
epic
(detector)
[DD4hep]
IRT
evaluation
(benchmarks)
irt
(library)
[standalone]
Data Model
EDM4hep
eicd
[PODIO]
temporary (?)
dependence
temporary
dependence
Reconstruction
IRT Plugin (?)
TO DO
EICrecon
[JANA2]
Geometry
Service
Benchmarks

IRT Integration Status

- **Connection to geometry**
  - Short term Backdoor: embed IRT code with geometry description
    - produce auxiliary optics config file, for IRT code, bypassing reconstruction framework
    - not sustainable for the future, won't scale, makes geometry depend on reconstruction
  - Long term approach: access the geometry from the reconstruction framework geometry service

- **Refactoring for upstream data model changes**
  - Moving toward EDM4hep data model requires us to update
  - Example: object indices replaced by relations
  - Refactoring needed in IRT benchmark code
  - We also propose additional components and datatypes: https://eicweb.phy.anl.gov/EIC/eicd/-/merge_requests/70

- **Integrate with Reconstruction Framework EICrecon (JANA2)**
  - Await more stability of EICrecon
  - Plugin to interface with standalone IRT library?
    - Read geometry service
    - Build IRT objects (surfaces etc.)
    - Convert dRICH hits to Cherenkov angle, PID hypothesis, etc.

- **Benchmarks → performance plots**

Automated Parameter Variation

https://github.com/c-dilks/drich-dev/pull/5

`scripts/vary_params.rb`

- Input user configuration:
  - Which compact file parameters to vary, and how to vary them
  - Fixed parameter values (which differ from the default)
  - Derived parameters, which depend on varied parameter values
  - Simulation pipeline – the code you want to run for each variant (shell commands)

- Execution
  - Takes the “product” of all possible variants
  - Calculates derived parameters for each variant
  - Generates dRICH compact files for each variant
  - Runs simulation pipelines, multi-threaded, one thread per variant

- Outputs for each variant:
  - Simulation pipeline output, as well as logs for stdout and stderr
  - Info files, listing the variant’s parameters
  - Compact files, config files, etc.

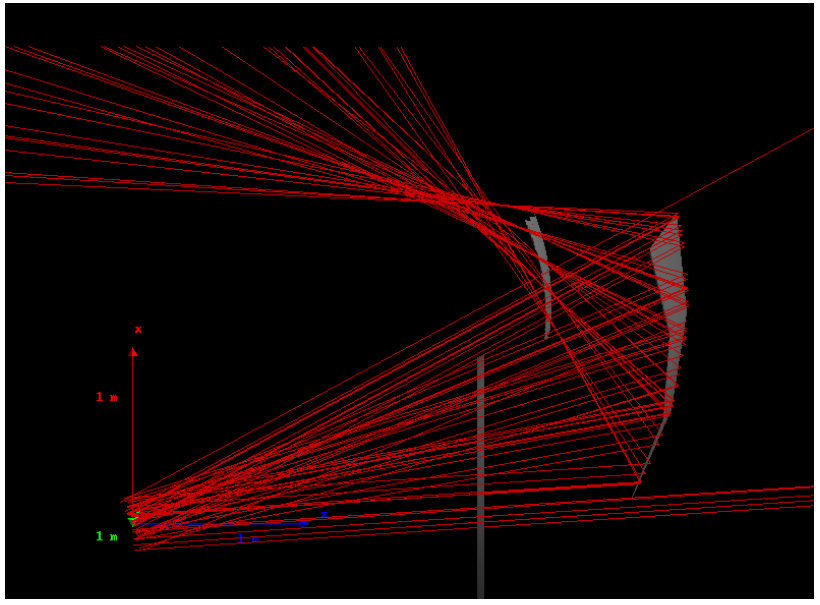
Can we use this or something similar to connect Machine Learning needs to simulation jobs?

- Ruby: dynamic scripting; excellent for automation and glue code
- Pycall.rb gem provides seamless access to Python libraries
- Added helper script to setup local Ruby environment and additional gems

Optics Tuning

Current Prod Version

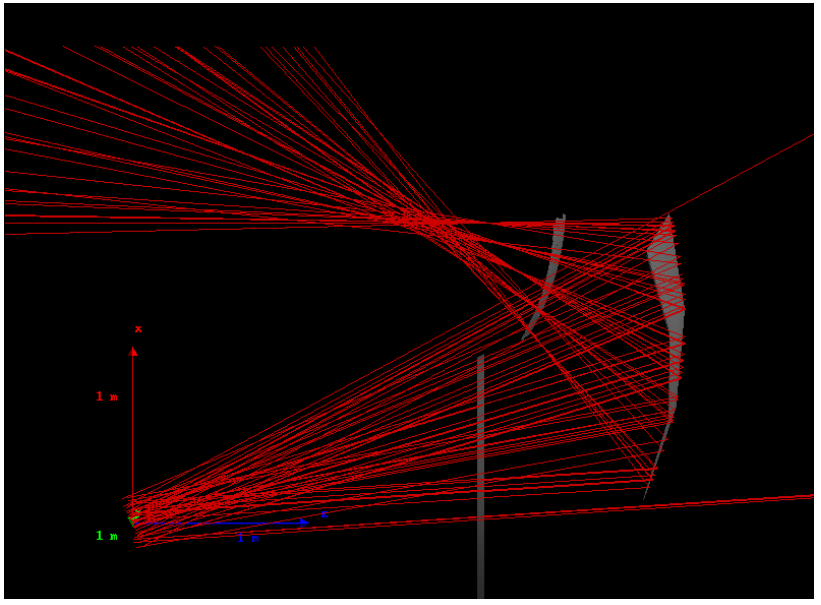
epic: main (691f7a8)



- High θ misses sensors
- Low θ has large angle of incidence on sensors

Updated Version (very very preliminary!)

epic: drich-optics-7-27 (a0ca649)

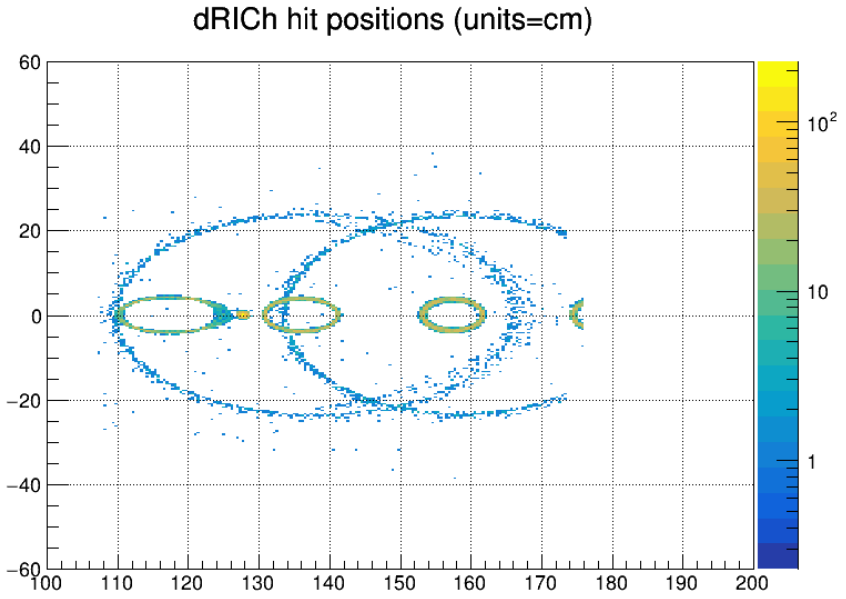


- Angles of incidence are smaller for all θ
- Parallel-to-point foci need more steering

Optics Tuning

Current Prod Version

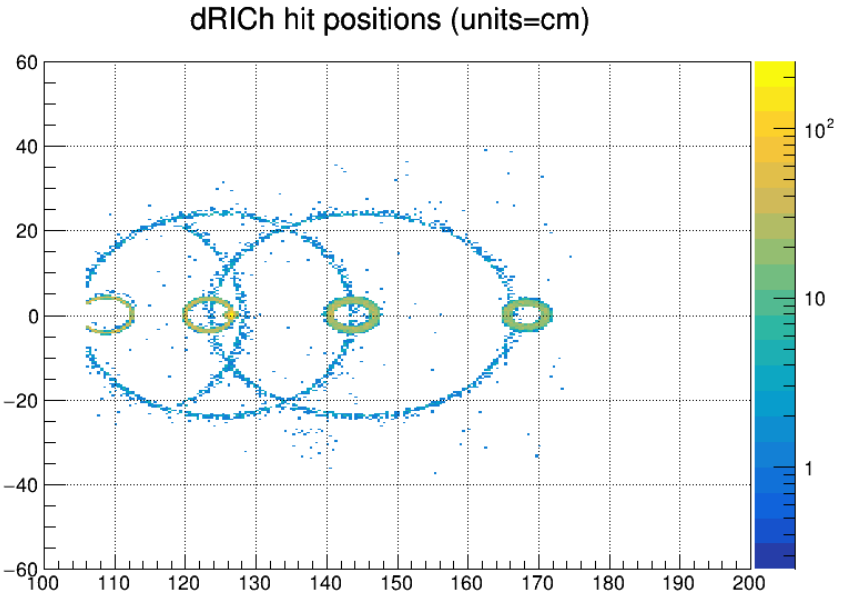
epic: main (691f7a8)



- High θ misses sensors
- Low θ has large angle of incidence on sensors

Updated Version (very very preliminary!)

epic: drich-optics-7-27 (a0ca649)



- Angles of incidence are smaller for all θ
- Parallel-to-point foci need more steering

Next Steps

Pending merge requests and ongoing work:
https://github.com/c-dilks/drich-dev/blob/main/doc/branches.md

- Improve the optics – in progress
- Revive IRT and add benchmarks → baseline algorithm – in progress
- Add and test additional pattern recognition / PID algorithms → help wanted
- Update sensor materials (https://eicweb.phy.anl.gov/EIC/detectors/ecce/-/issues/12) – in progress
- Determine what we can do to support machine learning optimization techniques
- Model focal region, for ideal sensor placement – in progress
- Explore Dual / multi-mirror configurations (see https://eicweb.phy.anl.gov/EIC/detectors/athena/-/merge_requests/260)
- Automate generation of material property tables (see https://github.com/cisbani/dRICh/blob/main/share/source/g4dRIChOptics.hh)
- Align optics with proton beam?