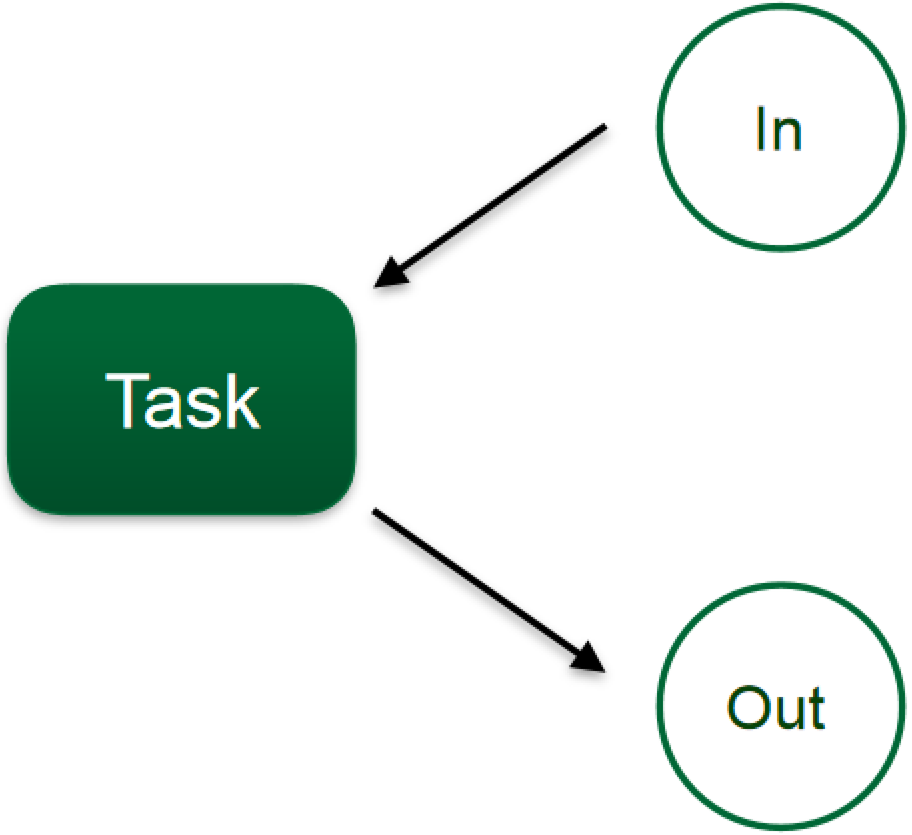
ATTPCROOTv2

Yassid Ayyad

What's ATTPCROOT

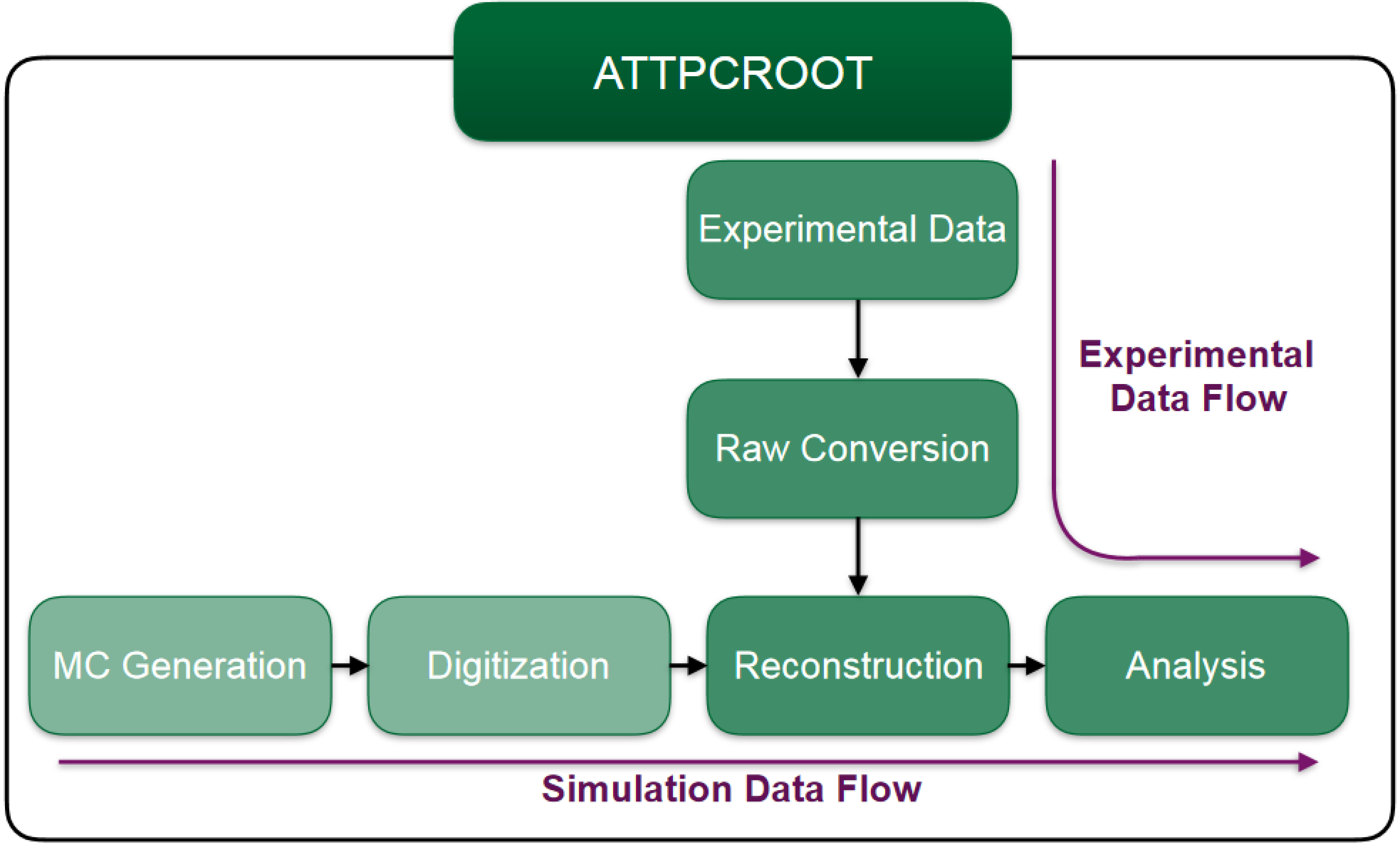
- Modular data analysis framework for Active Target Time Projection Chambers and Solenoidal spectrometers.
- Based on FairRoot package (developed at FAIR).
- Contains a collection of scientific libraries used in nuclear physics: ROOT, Geant4, physics generators, management libraries.
- Written in C++ following the latest standards.
- "User-friendly": analysis steered by ROOT macros.
- Support from our small community.

The basics



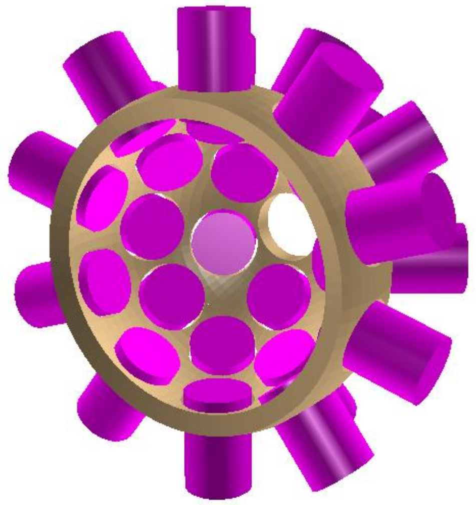
- ASCII
- GRAW
- ROOT
- An object on memory (TClonesArray*)

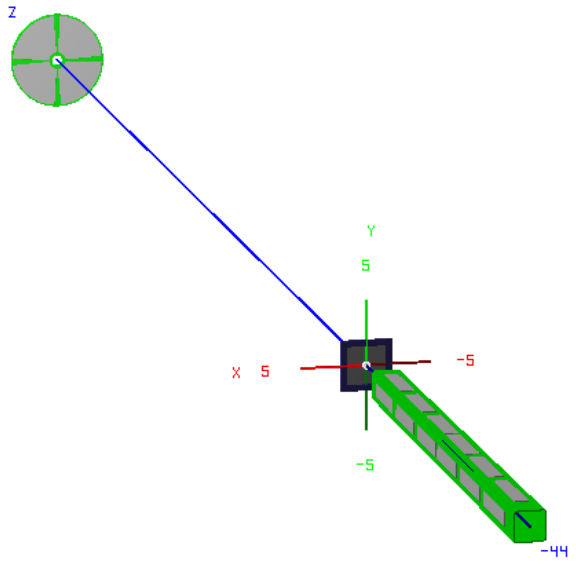
- ROOT
- An object on memory (TClonesArray*)


ATTPCROOT
Experimental Data
Raw Conversion
Experimental Data Flow
MC Generation
Digitization
Reconstruction
Analysis
Simulation Data Flow

Simulation tasks

- User defines a geometry that it is stored in a ROOT file: TGeo geometry and list of materials. It can (and it should be) as complex as needed.
- Passive elements can be added: magnets, pipes…
- User chooses magnetic field map.
- User chooses simulation engine and parameters (both, Mc and detector). Interface with Geant4 done through VMC.
- User defines a list of physics generators.
- Complete simulation is done in two steps: MC and digitalization. For TPC-like detectors, the user has to provide the geometry and response of the pad plane.





Monte Carlo Macro

```
void Be10He4_sim(Int_t nEvents = 10000, TString mcEngine = "TGeant4")
{

  TString dir = getenv("VMCWORKDIR");

  // Output file name
  TString outFile ="./data/attpcsim.root";

  // Parameter file name
  TString parFile="./data/attpcpar.root";

  // -----   Timer   --------------------------------------------------------
  TStopwatch timer;
  timer.Start();
  // ------------------------------------------------------------------------

  //gSystem->Load("libAtGen.so");

  ATVertexPropagator* vertex_prop = new ATVertexPropagator();


  // -----   Create simulation run   ----------------------------------------
  FairRunSim* run = new FairRunSim();
  run->SetName(mcEngine);              // Transport engine
  run->SetOutputFile(outFile);          // Output file
  FairRuntimeDb* rtdb = run->GetRuntimeDb();
  // ------------------------------------------------------------------------
```

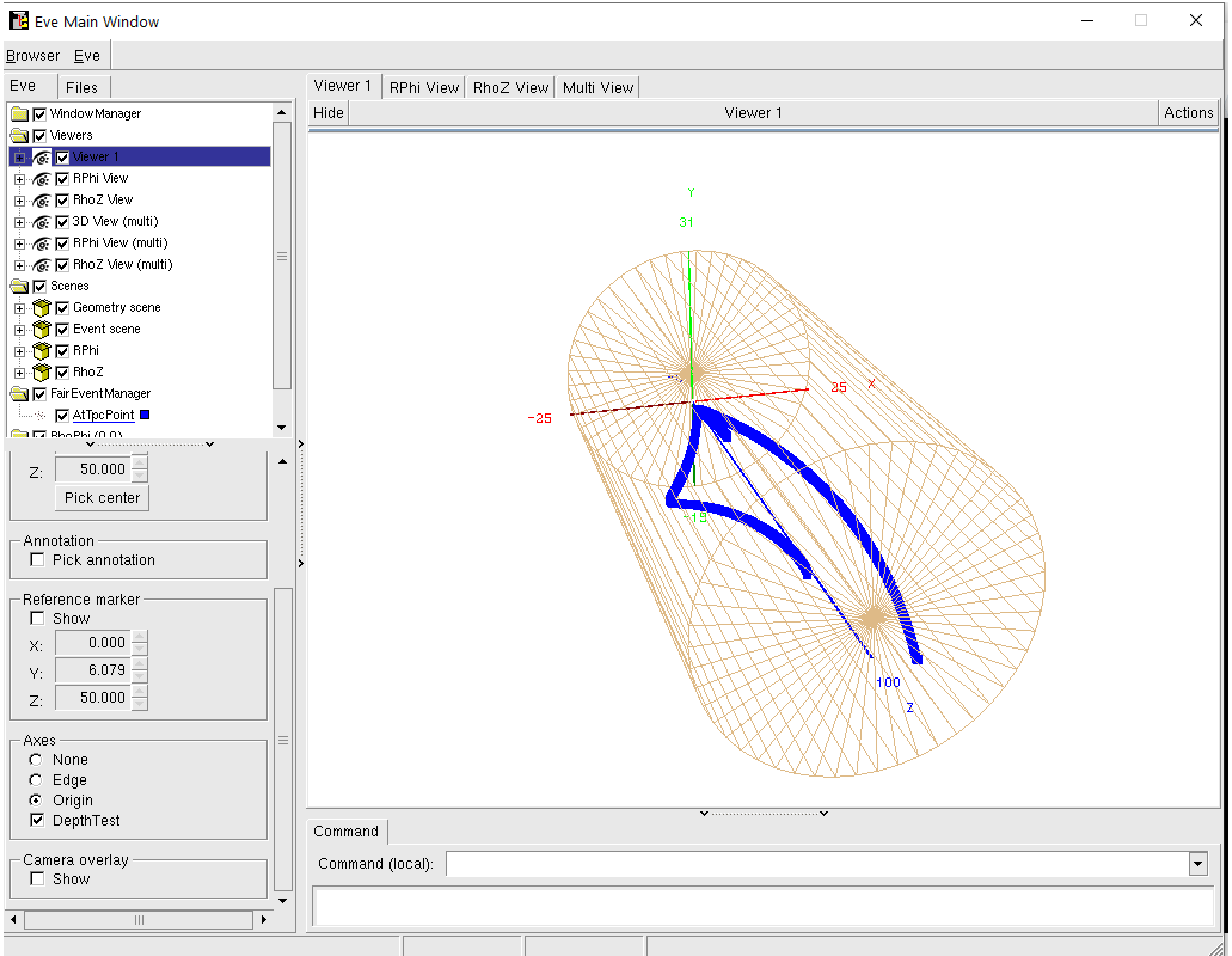
Monte Carlo Macro

```
// -----   Create media   -------------------------------------------------
run->SetMaterials("media.geo");       // Materials
// ------------------------------------------------------------------------

// -----   Create geometry   ----------------------------------------------

FairModule* cave= new AtCave("CAVE");
cave->SetGeometryFileName("cave.geo");
run->AddModule(cave);

//FairModule* magnet = new AtMagnet("Magnet");
//run->AddModule(magnet);

/*FairModule* pipe = new AtPipe("Pipe");
run->AddModule(pipe);*/

FairDetector* ATTPC = new AtTpc("ATTPC", kTRUE);
ATTPC->SetGeometryFileName("ATTPC_He1bar.root");
//ATTPC->SetModifyGeometry(kTRUE);
run->AddModule(ATTPC);

// ------------------------------------------------------------------------


  // -----   Magnetic field   -------------------------------------------
  // Constant Field
  AtConstField  *fMagField = new AtConstField();
  fMagField->SetField(0., 0. ,20. ); // values are in kG
  fMagField->SetFieldRegion(-50, 50,-50, 50, -10,230); // values are in cm
                      //  (xmin,xmax,ymin,ymax,zmin,zmax)
  run->SetField(fMagField);
  // --------------------------------------------------------------------
```

```
// -----   Create PrimaryGenerator   -------------------------------------
FairPrimaryGenerator* primGen = new FairPrimaryGenerator();




                // Beam Information
                Int_t z = 4;  // Atomic number
                Int_t a = 10; // Mass number
                Int_t q = 0;   // Charge State
                Int_t m = 1;   // Multiplicity  NOTE: Due the limitation of the TGenPhaseSpace accepting only pointers/arrays the maximum multiplicity has been set
                Double_t px = 0.000/a;  // X-Momentum / per nucleon!!!!!!
                Double_t py = 0.000/a;  // Y-Momentum / per nucleon!!!!!!
                Double_t pz = 0.86515/a;  // Z-Momentum / per nucleon!!!!!!
                Double_t BExcEner = 0.0;
                Double_t Bmass =10.013533818;
                Double_t NomEnergy = 40.0;



                ATTPCIonGenerator* ionGen = new ATTPCIonGenerator("Ion",z,a,q,m,px,py,pz,BExcEner,Bmass,NomEnergy);
                ionGen->SetSpotRadius(0,-100,0);
                // add the ion generator

                primGen->AddGenerator(ionGen);

                //primGen->SetBeam(1,1,0,0); //These parameters change the position of the vertex of every track added to the Primary Generator
                // primGen->SetTarget(30,0);
```

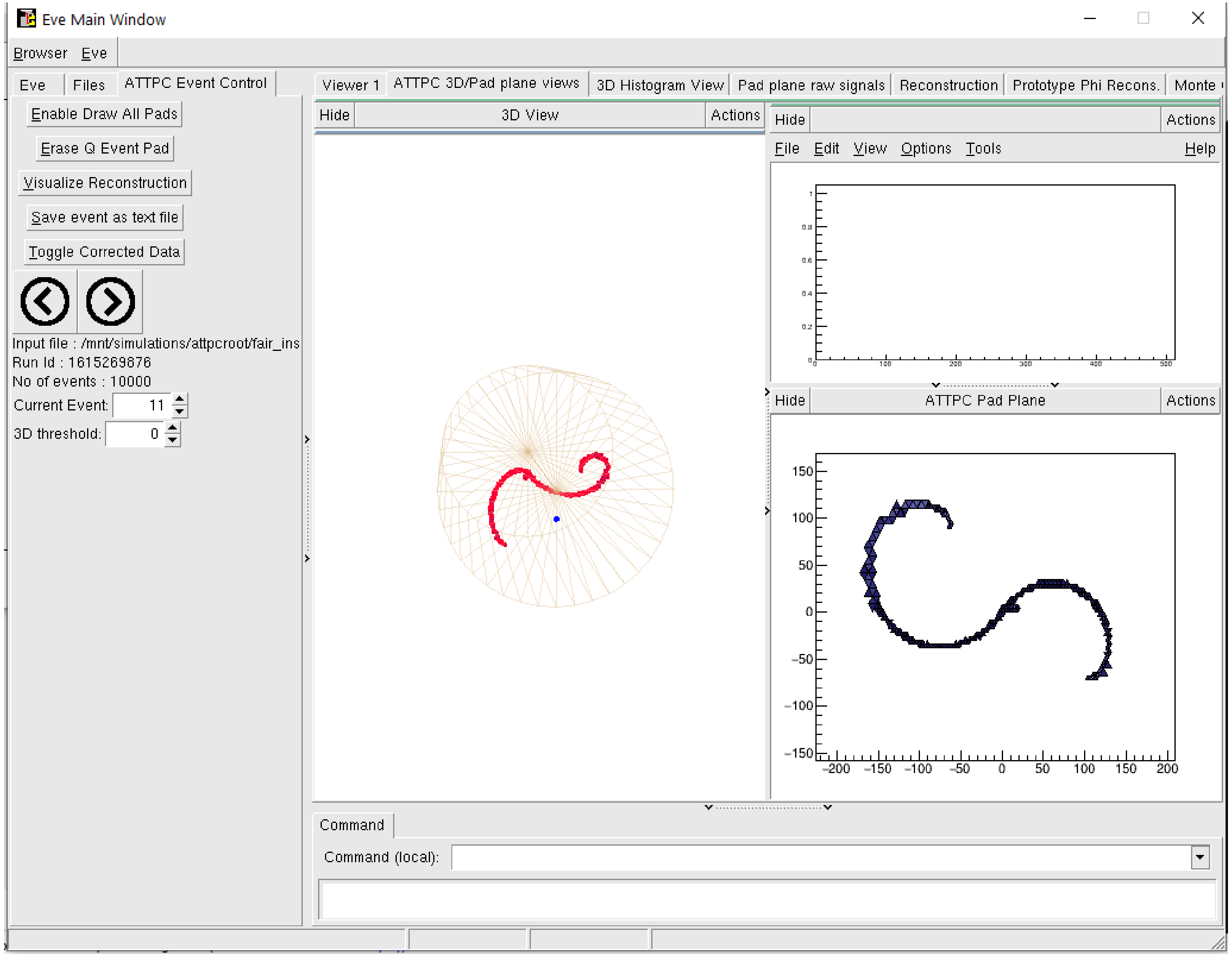
```
        //--- Scattered -----
        Zp.push_back(4); // 12Be TRACKID=1
        Ap.push_back(10); //
        Qp.push_back(0);
        Pxp.push_back(0.0);
        Pyp.push_back(0.0);
        Pzp.push_back(0.0);
        Mass.push_back(10.013533818);//uma
        ExE.push_back(0.0);


        // ---- Recoil -----
        Zp.push_back(2); // p  TRACKID=2
        Ap.push_back(4); //
        Qp.push_back(0); //
        Pxp.push_back(0.0);
        Pyp.push_back(0.0);
        Pzp.push_back(0.0);
        Mass.push_back(4.00260325415);//uma
        ExE.push_back(0.0);//In MeV


        Double_t ThetaMinCMS = 0.0;
        Double_t ThetaMaxCMS = 90.0;


    ATTPC2Body* TwoBody = new ATTPC2Body("TwoBody",&Zp,&Ap,&Qp,mult,&Pxp,&Pyp,&Pzp,&Mass,&ExE,ResEner,ThetaMinCMS,ThetaMaxCMS);
    primGen->AddGenerator(TwoBody);


    run->SetGenerator(primGen);
```

eventDisplay.C

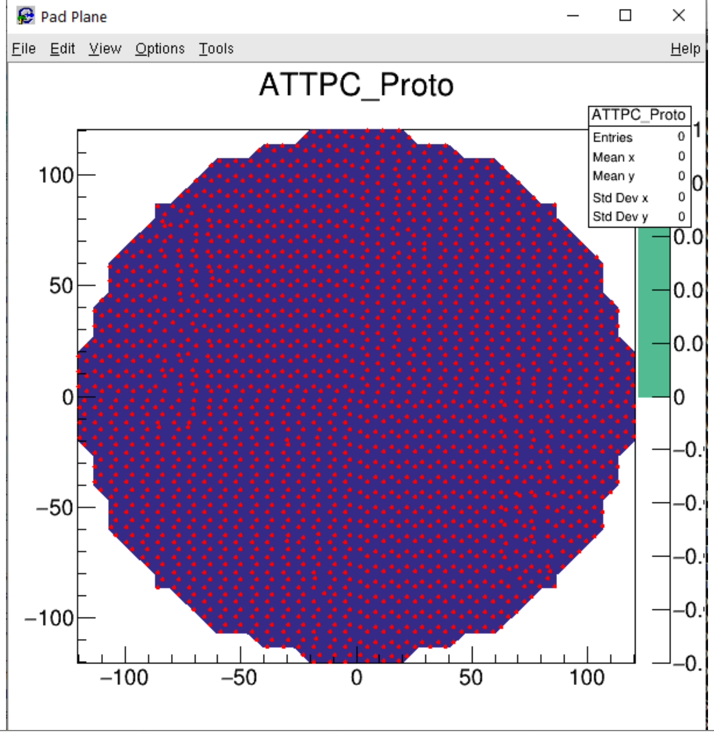


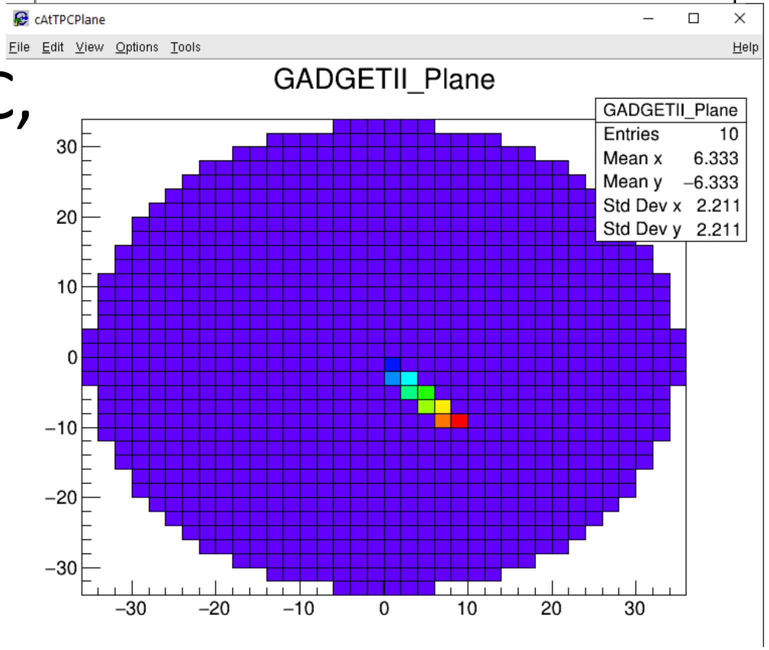
```
    ATClusterizeTask* clusterizer = new ATClusterizeTask();
    clusterizer -> SetPersistence(kFALSE);

    ATPulseTask* pulse = new ATPulseTask();
    pulse -> SetPersistence(kTRUE);
    pulse -> SetSaveMCInfo();

    ATPSATask *psaTask = new ATPSATask();
    psaTask -> SetPersistence(kTRUE);
    psaTask -> SetThreshold(10);
    //psaTask -> SetPSAMode(1); //NB: 1 is ATTPC - 2 is pATTPC
    psaTask -> SetPSAMode(1); //FULL mode
    //psaTask -> SetPeakFinder(); //NB: Use either peak finder of maximum finder but not both at the same time
    psaTask -> SetMaxFinder();
    psaTask -> SetBaseCorrection(kFALSE); //Directly apply the base line correction to the pulse amplitude to correct for the mesh induction. If false the correcti
    psaTask -> SetTimeCorrection(kFALSE); //Interpolation around the maximum of the signal peak

    ATPRATask *praTask = new ATPRATask();
    praTask->SetPersistence(kTRUE);

    /*ATTriggerTask *trigTask = new ATTriggerTask();
    trigTask  ->  SetAtMap(mapParFile);
    trigTask  ->  SetPersistence(kTRUE);*/


fRun -> AddTask(clusterizer);
fRun -> AddTask(pulse);
// fRun -> AddTask(psaTask);
// fRun -> AddTask(praTask);
// fRun -> AddTask(trigTask);

// __ Init and run ___________________________________

fRun -> Init();
fRun->Run(0, 20);
```

run_eve.C (simulation and experimental data)



- TPC’s maps are managed by a virtual Map class. The pad plane uses TH2Poly histogram to draw and simulate.
- Energy loss -> Electrons -> Avalanche. Drift electrons are propagated one by one taking into account the operational parameters of the detector.
- Supported detectors for digitalization: pATTPC, ATTPC, SOLARIS Si array+Apollo, GADGET, SpecMAT.
- Signal generation adapted to GET electronics + Micromegas. Soon resistive micromegas.





Next meeting: Analysis and reconstruction…