Tutorial on DSimu and DAna

Tong Sun, Xuliang Zhu

2023.4.8

李政道研究所
TSUNG-DAO LEE INSTITUTE

李政道研究所
TSUNG-DAO LEE INSTITUTE

目　　录
contents

PART 01

Get started with DSS

Please enter a title

TSUNG-DAO LEE INSTITUTE

A Quick Overview of DSS

DSimu: the simulation program based on Geant4 and ROOT

DAna: framework for analysis and reconstruction

DDis: the event display tool

DPlot: basic plotting program for quick plot, based on ROOT

Installation of DSS

Download DSS from GitLab

```
bash$ git clone git@code.ihep.ac.cn:darkshine/darkshine-simulation.git
```

Before installing, if you are using your own machine, several dependencies need to be checked.

- C++17
- Geant4 10.06
- ROOT 6 (>=6.20)
- gsl
- yaml-cpp
- nlohmann/json

Or, if you are using clusters with CVMFS installed, you can directly source the LCG environment:

```
bash$  source /cvmfs/sft.cern.ch/lcg/views/LCG_97rc4python3/x86_64-centos7-gcc9-opt/setup.sh
```

How to build DSS

Using CMake to Build DSS

```
cd darkshine-simulation   # <source-directory>
mkdir build
cd build
cmake -DCMAKE_INSTALL_PREFIX=<your-install-directory> ../
make -j100  # Just do it!
make install
cd ..
```

Write a file to export environment variables, i.e. setup.sh

```
source /cvmfs/sft.cern.ch/lcg/views/LCG_97rc4python3/x86_64-centos7-gcc9-opt/setup.sh
DSS_DIR=<your-install-directory>
export PATH=${DSS_DIR}/bin:${PATH}
export LD_LIBRARY_PATH=${DSS_DIR}/lib:${LD_LIBRARY_PATH}
```

Then, source this file.

```
source setup.sh
```

How to compile a changed package

if you just edit the code

```
bash$ cd <build directory>
bash$ make && make install
```

if you add or remove files

```
bash$ cd <build directory>
bash$ cmake
bash$ make && make install
```

PART 02

Guide on DSimu

TSUNG-DAO LEE INSTITUTE

Running a job of simulation

- Bach mode by reading a YAML file

```
bash$ cd build
bash$ DSimu –y default.yaml
```

- Interactive mode via a Graphical User Interface

```
bash$ DSimu -g
```

YAML config file:

- Geant4 General Settings
- Global Variables
- Magnetic field
- Root Manager Settings
- DEvent Collection
- Event Biasing
- Event Filters
- Optical Options
- Detector Geometry

After simulation complicated, <dp_out>.root can found under the output directory.

Guide on DSimu

Bash mode via YAML file

YAML

- a human friendly data serialization standard for all programming languages
- a fixed indentation scheme to represent relationships between data layers

You can also use inline syntax for lists and dictionaries

```
episodes: [1, 2, 3, 4, 5, 6, 7]
best-jedi: {name: Obi-Wan, side: light}
```

YAML Syntax

- Scalars

Scalars are ordinary values: number, string, bool

```
visible_decay: false
# decay channel: [ee or mumu]
dp_decay_channel: "ee"
dp_eplsion: 1e-3
```

- Lists

Lists are collections of elements: starts with a dash and a space

```
tag_Size_Tracker:
  - [ 10.05 , "cm", 20 , "cm", 0.1 , "mm" ]
  - [ 10.05 , "cm", 20 , "cm", 0.1 , "mm" ]
  - [ 10.05 , "cm", 20 , "cm", 0.1 , "mm" ]
  - [ 10.05 , "cm", 20 , "cm", 0.1 , "mm" ]
  - [ 10.05 , "cm", 20 , "cm", 0.1 , "mm" ]
  - [ 10.05 , "cm", 20 , "cm", 0.1 , "mm" ]
```

- Dictionaries

Dictionaries are collections of key: value mappings

```
Biasing:
  if_bias: false
  if_bias_target: false
  if_bias_ECAL: false
```

Guide on DSimu

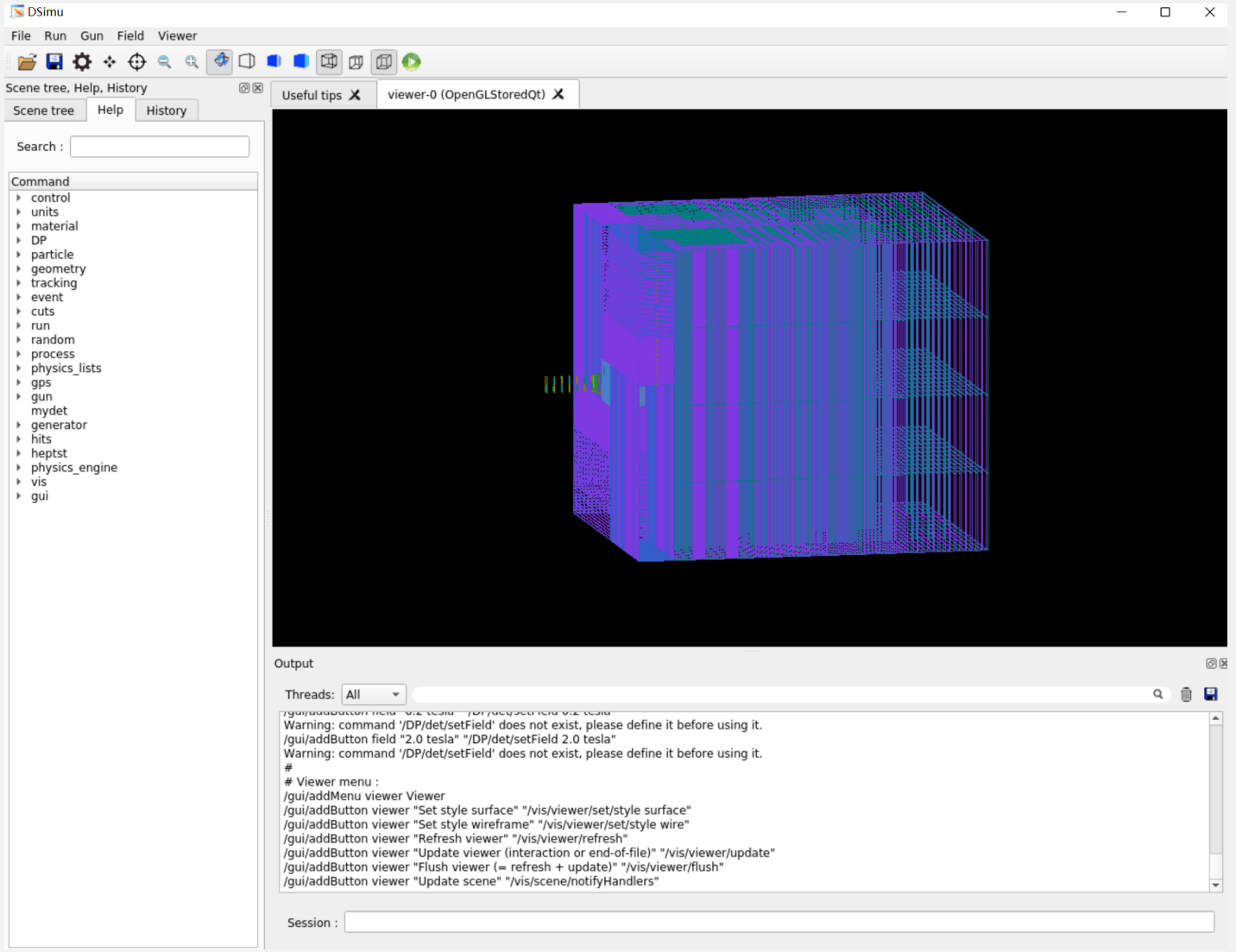
Visualization

via a Graphical User Interface
Note: Copy the gui.mac, init_vis.mac, vis.mac to the same directory

```
bash$ DSimu -g
```

Open a session with a command box

- UI commands
- shell commands "exit, ls, cd… "



Guide on DSimu

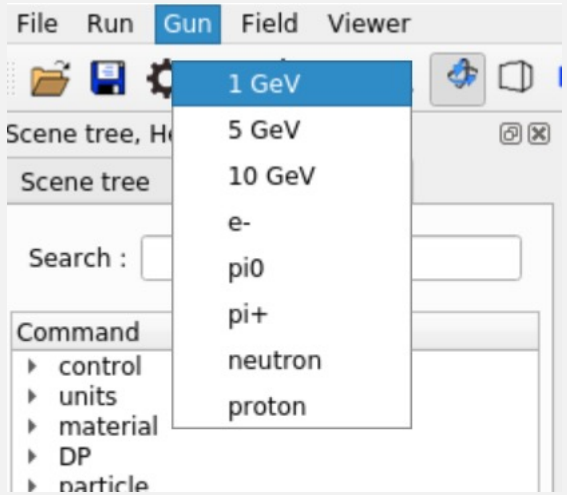
UI commands

Build-in commands:

/control/	UI control commands.
/units/	Available units.
/profiler/	Profiler controls.
/gui/	UI interactors commands.
/tracking/	TrackingManager and SteppingManager control commands.
/geometry/	Geometry control commands.
/process/	Process Table control commands.
/particle/	Particle control commands.
/event/	EventManager control commands.
/cuts/	Commands for G4VUserPhysicsList.
/run/	Run control commands.
/random/	Random number status control commands.
/material/	Commands for materials
/physics_lists/	commands related to the physics simulation engine.
/analysis/	analysis control
/vis/	Visualization commands.
/hits/	Sensitive detectors and Hits
/physics_engine/	...Title not available...
/gun/	Particle Gun control commands.

example: gui.mac

```
/gui/addMenu gun Gun
/gui/addButton gun "1 GeV"    "/gun/energy 1 GeV"
/gui/addButton gun "5 GeV"    "/gun/energy 5 GeV"
/gui/addButton gun "10 GeV"   "/gun/energy 10 GeV"
/gui/addButton gun "e-"       "/gun/particle e-"
/gui/addButton gun "pi0"      "/gun/particle pi0"
/gui/addButton gun "pi+"      "/gun/particle pi+"
/gui/addButton gun "neutron"  "/gun/particle neutron"
/gui/addButton gun "proton"   "/gun/particle proton"
```



PART 03

Guide on DAna

TSUNG-DAO LEE INSTITUTE

Guide on DAna

How to run

Create config file

```
bash$ DAna –x > config.txt
```

Run

```
bash$ DAna -c config.txt
```

```
############################
###
### Example Config File
###
############################

### Basic Settings
InputFile       = dp_out.root
InputGeoFile    = dp_out.root
OutputFile      = dp_ana.root
RunNumber       = 0
EventNumber     = -1
SkipNumber      = 0

### Verbosity Settings
AlgoManager.Verbose           = 0
EventReader.Verbose           = 2
EventStoreAndWriter.Verbose   = 0
MemoryCheck.Verbose           = 0

### Algorithm List
###
# Digitizer: Digitizer for Calorimeter(ECAL) with optical process
# MCTruthAnalysis: MC Truth Analysis
# RecECAL: ECAL Reconstruction Processor
# Tracking: Tracking by Yi-Fan Zhu
# CutFlowAnalysis: None
###
Algorithm.List = Digitizer MCTruthAnalysis RecECAL Tracking
```

```
RecECAL.Advance = 2  # Advanced analysis level
RecECAL.E_n_fraction = 20  # the n-th large E fraction
RecECAL.SaveTrackInfo = 0  # SaveTrackInfo
RecECAL.SaveTruthInfo = 0  # SaveTruthInfo(from MCparticle)
RecECAL.SkipEmpty = 0  # Skip Empty Hits
RecECAL.Verbose = 0  # Verbosity Variable
RecECAL.ECollectionToUse = ECAL_FS0,ECAL_FS1,ECAL_FS2,ECAL_FS3,ECAL
RecECAL.HCollectionToUse = FS0  # Calorimeter (HCAL) Collection to

Tracking.Rec_fit_method = 1  # Specify fit method: 0, no fine fit;
Tracking.Tag_fit_method = 1  # Specify fit method: 0, no fine fit;
Tracking.clean = 0  # Clean mode: no truth information
Tracking.if_smear = 1  # If smear hits in strip structure
Tracking.if_strip = 1  # If use strip structures in trackers
Tracking.verbose = 0  # Verbose
Tracking.con_field = -1.5  # Const magnet field
```

Reconstruction & Analysis

DAna is a software framework that will call user-defined functions, named Processor.

Processor

- Plug-in modules that can access the event raw data (DEvent), to implement some functionality

Current Processors

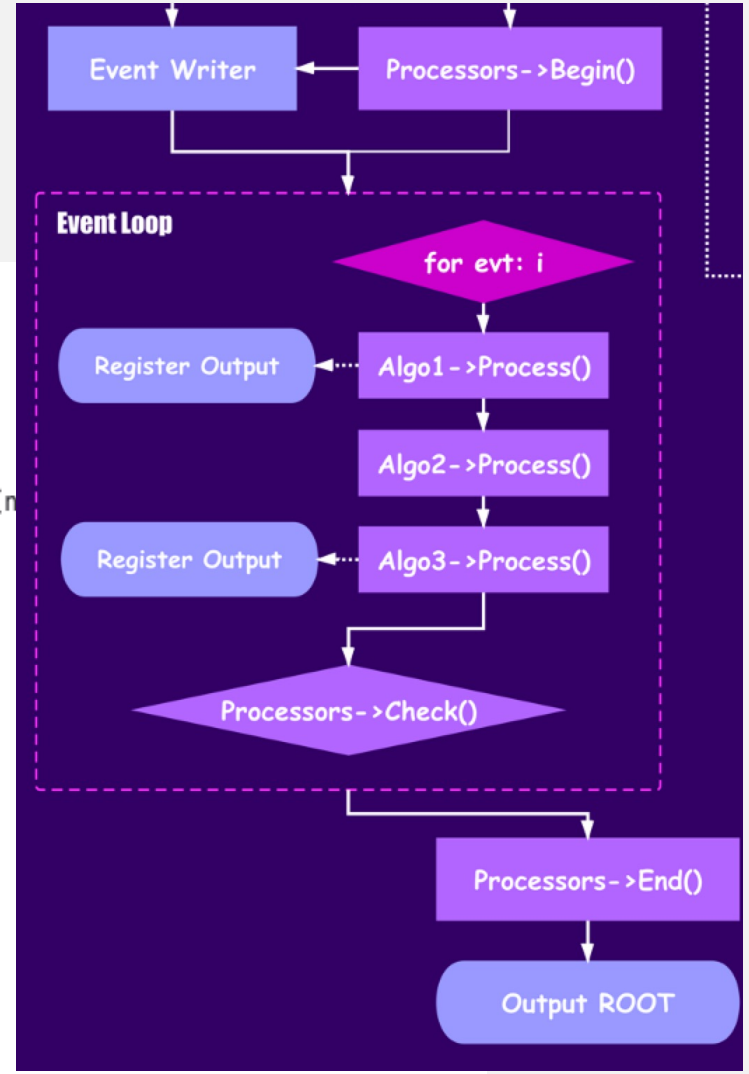
- Digitizer
- MC Truth Analysis
- RecECAL
- TrackingProcessor

Example processor

- Algorithms/ExampleProcessor/include/Algo/ExampleProcessor.h
- Algorithms/ExampleProcessor/src/ExampleProcessor.cpp

```
12  using namespace std;
13
14  class ExampleProcessor : public AnaProcessor {
15  public:
16      // No need to change anything here
17      // Must initialized with Name
18      explicit ExampleProcessor(string name, shared_ptr<EventStoreAndWriter> evtwrt) : AnaProcessor(std::move(n
19      ~ExampleProcessor() override = default;
20
21      void Begin() override;
22
23      void InitEvt() override
24      {
25          Output_Var1 = 0;
26          Output_Var2 = 0.;
27      }
28
29      void ProcessEvt(AnaEvent* evt) override;
30
31      void CheckEvt(AnaEvent* evt) override;
32
33      void End() override;
34
35      // Define some functions here if necessary
```

typedef DEvent AnaEvent;



Practice

- **Write a Processor to read truth particles entering ECAL surface.**

1. Register ExampleProcessor (DP_ana/src/ControlManager.cpp)

```
algo->RegisterAnaProcessor(shared_ptr<ExampleProcessor>(new ExampleProcessor("ExampleProcessor",
EvtWrt)));
```

2. Modify the ExampleProcessor (Algorithms/ExampleProcessor/src/ExampleProcessor.cpp)
   1. loop through truth information
      - particles entering ECAL can be get by: evt->getTruthInfo()->getStatesInECAL();
      - See Usage of DTruthState:
   2. Save x, y, z, px, py, pz, E, pdg to ntuple

3. Compile & Run
   1. make && make install
   2. DSimu –y default.yaml –b 100
   3. DAna –x > config.txt
   4. DAna –c config.txt

```
map <pair<int, int>, pair<DTruthState *, DTruthState *>> DTruth::getStatesInECAL(float E_min) {
    return getStatesInCalorimeter(DTruthDetPV::ECAL, E_min);
}
```

```
map <pair<int, int>, pair<DTruthState *, DTruthState *>>
DTruth::getStatesInCalorimeter(DTruth::DTruthDetPV DetPV, float E_min) {
    /*
     * Output format:
     *  map<{trackID, PDG}, {prev_state, post_state}>
     *  prev_state: state just before entering the detector PV
     *  post_state: state just after entering the detector PV
     */
```

Guide on DAna

Practice

```
void ExampleProcessor::Begin() {
    // Register Output Variable
    EvtWrt->RegisterOutVariable("InECAL_X", &InECAL_X);
    EvtWrt->RegisterOutVariable("InECAL_Y", &InECAL_Y);
}
```

```
void ExampleProcessor::ProcessEvt(AnaEvent *evt) {
    // read DTruth
    for (auto const & id_states: evt->getTruthInfo()->getStatesInECAL()) {
        // get post_state
        DTruthState* state = id_states.second.second;
        InECAL_X.emplace_back(state->vertex[0]);
        InECAL_Y.emplace_back(state->vertex[1]);
    }
}
```

```
    void InitEvt() override
    {
        InECAL_X.clear();
        InECAL_Y.clear();
    }
```

谢 谢 ！