Update on fast simulation in Geant4

Anna Zaborowska

SFT Simulation R&D meeting

April 28, 2020

In short

Recent work:

- towards updated tunable EM shower parameterisation
  & inference of previously-trained ML generative models within Geant4.

In short

Recent work:

- towards updated tunable EM shower parameterisation
  & inference of previously-trained ML generative models within GEANT4.

- started with a closer look at fast simulation in GEANT4:
  - `source/processes/parameterisation/` general
  - `source/parametrisations/`only GFlash

In short

Recent work:

- towards updated tunable EM shower parameterisation
  & inference of previously-trained ML generative models within Geant4.

- started with a closer look at fast simulation in Geant4:
  - `source/processes/parameterisation/` general
  - `source/parametrisations/`only GFlash

- create tools that are common for (many) fast simulation models (some based on GFlash)

In short

Recent work:

- towards updated tunable EM shower parameterisation
  & inference of previously-trained ML generative models within Geant4.

- started with a closer look at fast simulation in Geant4:
  - `source/processes/parameterisation/` general
  - `source/parametrisations/`only GFlash

- create tools that are common for (many) fast simulation models (some based on GFlash)

- not yet presented at Generic Processes WG

In short

Recent work:

- towards updated tunable EM shower parameterisation
  & inference of previously-trained ML generative models within GEANT4.
- started with a closer look at fast simulation in GEANT4:
  - `source/processes/parameterisation/` general
  - `source/parametrisations/`only GFlash
- create tools that are common for (many) fast simulation models (some based on GFlash)
- not yet presented at Generic Processes WG
- prototyped inside example application on gitlab, to be included as core source and an example;

Fast simulation in GEANT4

Fast simulation model is a process, with final state of particle described by `G4FastStep` (derived from `G4VParticleChange`).

Fast simulation in Geant4

Fast simulation model is a process, with final state of particle described by `G4FastStep` (derived from `G4VParticleChange`).

`G4VFastSimulationModel` is used to describe what happens to particle:

```
void SimplestFastSimModel::DoIt(const G4FastTrack& fastTrack, G4FastStep& fastStep)
{
  fastStep.KillPrimaryTrack();
  fastStep.SetPrimaryTrackPathLength(0.0);
  fastStep.SetTotalEnergyDeposited(fastTrack.GetPrimaryTrack()->GetKineticEnergy());
  [...] // how to deposit energy, CORE of fast simulation
}
```

Fast simulation in Geant4

Fast simulation model is a process, with final state of particle described by `G4FastStep` (derived from `G4VParticleChange`).

`G4VFastSimulationModel` is used to describe what happens to particle:

```
void SimplestFastSimModel::DoIt(const G4FastTrack& fastTrack, G4FastStep& fastStep)
{
  fastStep.KillPrimaryTrack();
  fastStep.SetPrimaryTrackPathLength(0.0);
  fastStep.SetTotalEnergyDeposited(fastTrack.GetPrimaryTrack()->GetKineticEnergy());
  [...] // how to deposit energy, CORE of fast simulation
}
```

[...] - can mean anything:

- produce secondary particles,
- alter particle's properties: energy, momentum, position (and go back to full sim),
- kill particle,
- deposit energy.

Fast simulation in Geant4

Fast simulation model is a process, with final state of particle described by `G4FastStep` (derived from `G4VParticleChange`).

`G4VFastSimulationModel` is used to describe what happens to particle:

```
void SimplestFastSimModel::DoIt(const G4FastTrack& fastTrack, G4FastStep& fastStep)
{
  fastStep.KillPrimaryTrack();
  fastStep.SetPrimaryTrackPathLength(0.0);
  fastStep.SetTotalEnergyDeposited(fastTrack.GetPrimaryTrack()->GetKineticEnergy());
  [...] // how to deposit energy, CORE of fast simulation
}
```

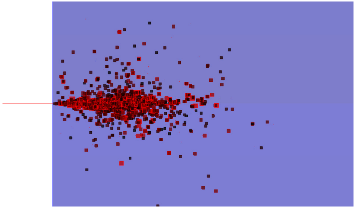
[...] - can mean anything:

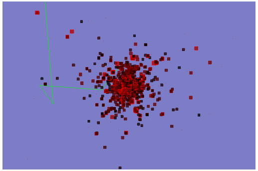
- produce secondary particles,
- alter particle's properties: energy, momentum, position (and go back to full sim),
- kill particle,
- deposit energy.

Most often deposit energy means place energy E in $\bar{r}$, no other information is passed to hit.
$\longrightarrow$ tools for energy deposition can be generalised.

Depositing energy in calorimeter

Full simulation

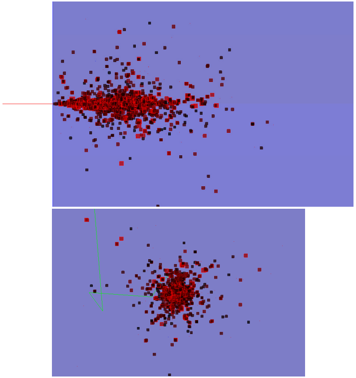




- 10 GeV e^-
- Showing deposits $E > 1\text{MeV}$

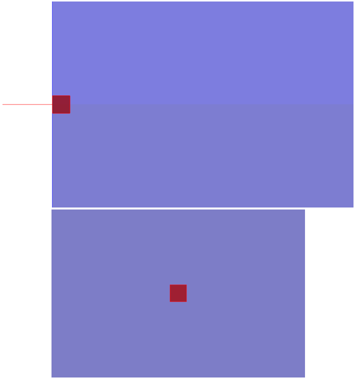
Depositing energy in calorimeter

Full simulation



- 10 GeV e$^-$
- Showing deposits $E > 1\text{MeV}$

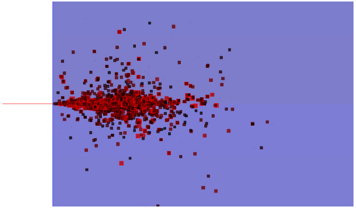
Fast simulation (simple)

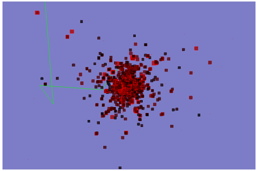


- deposit energy as described in `G4FastStep`
- to make more: user's responsibility

Depositing energy in calorimeter

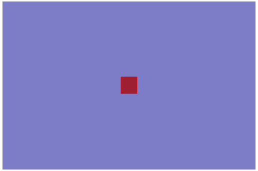
Full simulation





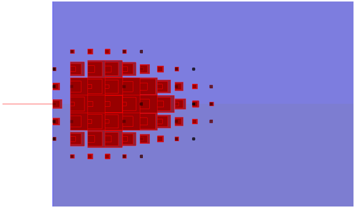
- 10 GeV e^-
- Showing deposits $E > 1$MeV

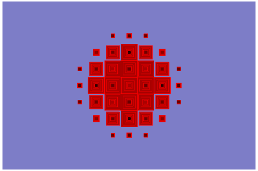
Fast simulation (simple)



- deposit energy as described in `G4FastStep`
- to make more: user's responsibility

Fast simulation (mesh)

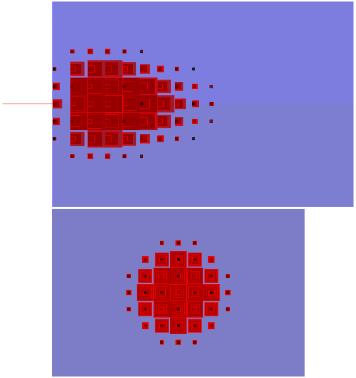




- generic tool looking for SD, placing hits mimicking full sim
- how energy is distributed still up to the user

Depositing energy in calorimeter

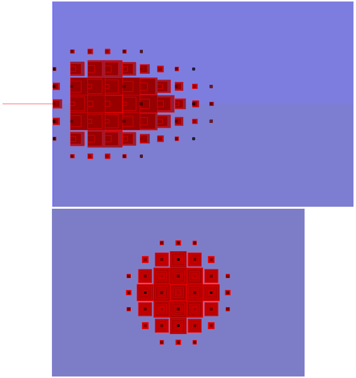
Fast simulation (mesh)



- Gamma (Gauss) distribution for longitudinal (transverse) profile, with arbitrary parameters (example only!)

Depositing energy in calorimeter

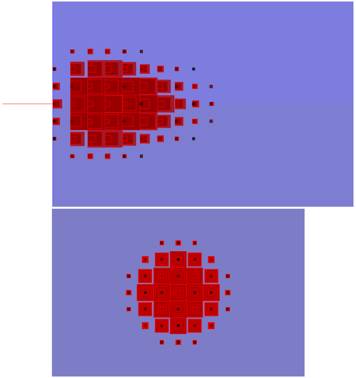
Fast simulation (mesh)



- Gamma (Gauss) distribution for longitudinal (transverse) profile, with arbitrary parameters (example only!)
- Can be used for parameterisations with previously tuned parameters (e.g. GFlash-like-mesh-based parameterisation)

Depositing energy in calorimeter

Fast simulation (mesh)



- Gamma (Gauss) distribution for longitudinal (transverse) profile, with arbitrary parameters (example only!)
- Can be used for parameterisations with previously tuned parameters (e.g. GFlash-like-mesh-based parameterisation)
- Can be used to deposit energy, e.g. from inference of NN models.
- Needed input:
  - any input to calculations (parameters, weights & model)
  - 3D tensor of values for energy deposits;
  - size and number of cells (XYZ);
  - input particle direction (for rotations);

Depositing energy in calorimeter

Status:

- Currently no general tool for depositing energy.
- GFlash implementation includes models, sensitive detector, ...
- They are all interlinked and connected to `GFlash*` classes.

Depositing energy in calorimeter

Status:

- Currently no general tool for depositing energy.
- GFlash implementation includes models, sensitive detector, …
- They are all interlinked and connected to `GFlash*` classes.

New developments:

- Simplify implementation of fast simulation models.
  - assumption: energy E_i is deposited in $\overline{r_i}$, $i \in 1, \ldots N$ (no other information)

Depositing energy in calorimeter

Status:

- Currently no general tool for depositing energy.
- GFlash implementation includes models, sensitive detector, …
- They are all interlinked and connected to `GFlash*` classes.

New developments:

- Simplify implementation of fast simulation models.
  - assumption: energy E_i is deposited in $\overline{r_i}$, $i \in 1, \ldots N$ (no other information)
- `G4FastHit` binding energy deposit and its position;

Depositing energy in calorimeter

Status:

- Currently no general tool for depositing energy.
- GFlash implementation includes models, sensitive detector, ...
- They are all interlinked and connected to `GFlash*` classes.

New developments:

- Simplify implementation of fast simulation models.
  - assumption: energy E_i is deposited in $\overline{r_i}$, $i \in 1, \ldots N$ (no other information)
- `G4FastHit` binding energy deposit and its position;
- `G4FastSimHitMaker` that keeps track of volumes, readout geometry, and calls sensitive detector to deposit energy;

Depositing energy in calorimeter

Status:

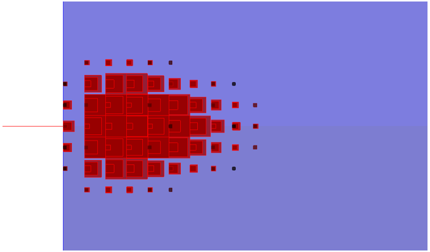
- Currently no general tool for depositing energy.
- GFlash implementation includes models, sensitive detector, ...
- They are all interlinked and connected to `GFlash*` classes.

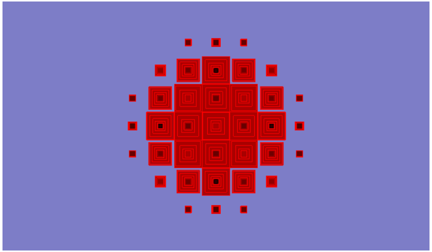
New developments:

- Simplify implementation of fast simulation models.
  - assumption: energy E_i is deposited in $\overline{r_i}$, $i \in 1, \ldots N$ (no other information)
- `G4FastHit` binding energy deposit and its position;
- `G4FastSimHitMaker` that keeps track of volumes, readout geometry, and calls sensitive detector to deposit energy;
- `G4VFastSimSensitiveDetector` is a base class for user's SD (additional to G4VSensitiveDetector) that includes method for processing hits based on `G4FastHit` (instead of the usual `G4Step`)

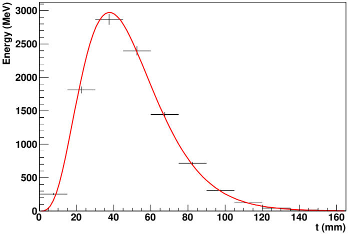
Simple analysis: fast simulation

In order to quickly verify fast simulation performance.
Implemented in event action (analysis of hit collection at the end of the event).

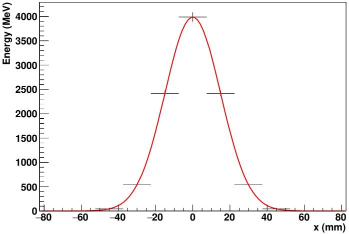




Longitudinal profile

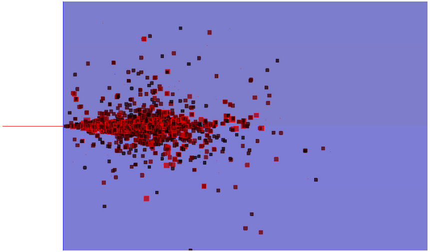


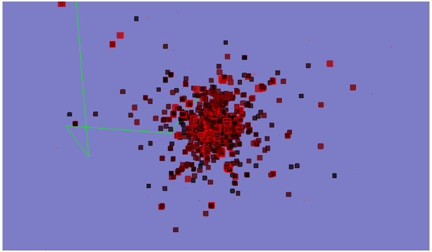
Transverse (x) profile



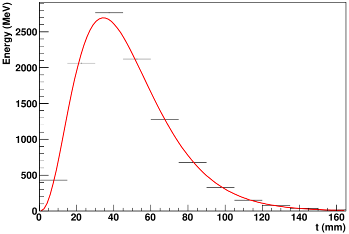
Simple analysis: full simulation

It uses the same class to describe hit, so can be applied to both full and fast simulation.

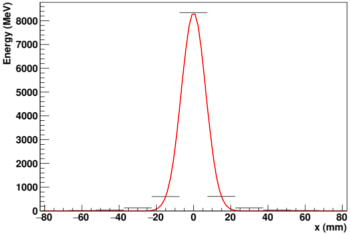




Longitudinal profile



Transverse (x) profile



Next steps

Energy deposits:

- Include rotations based on initial particle momentum;

Next steps

Energy deposits:

- Include rotations based on initial particle momentum;

Analysis (for parameterisation of showers):

- Create fast analysis to assess the quality of produced samples;
- Longitudinal and transverse average profiles, first and second moments;
- Deposited energy;

Next steps

Energy deposits:

- Include rotations based on initial particle momentum;

Analysis (for parameterisation of showers):

- Create fast analysis to assess the quality of produced samples;
- Longitudinal and transverse average profiles, first and second moments;
- Deposited energy;

Common models:

- Parameterisation model with different transverse, longitudinal functions (extension of GFlash model);

Next steps

Energy deposits:

- Include rotations based on initial particle momentum;

Analysis (for parameterisation of showers):

- Create fast analysis to assess the quality of produced samples;
- Longitudinal and transverse average profiles, first and second moments;
- Deposited energy;

Common models:

- Parameterisation model with different transverse, longitudinal functions (extension of GFlash model);
- Inference model that deposits energies from inference on pre-trained NN models;

Next steps

Energy deposits:

- Include rotations based on initial particle momentum;

Analysis (for parameterisation of showers):

- Create fast analysis to assess the quality of produced samples;
- Longitudinal and transverse average profiles, first and second moments;
- Deposited energy;

Common models:

- Parameterisation model with different transverse, longitudinal functions (extension of GFlash model);
- Inference model that deposits energies from inference on pre-trained NN models;
- Create robust messengers that allow to setup e.g. mesh size, input parameters,…