How to simulate ion CT with the new python-based Geant4 Monte Carlo software GATE 10

Nils Krah, CREATIS, Centre Léon Bérard, Lyon

Ion Imaging Workshop 2024, Vienna, Austria

Examples of Monte Carlo codes

Multi-purpose:

- Geant4, FLUKA, MCNP, …

Applications built on top of Geant4:

- TOPAS
- GATE version 9.x
- new GATE 10

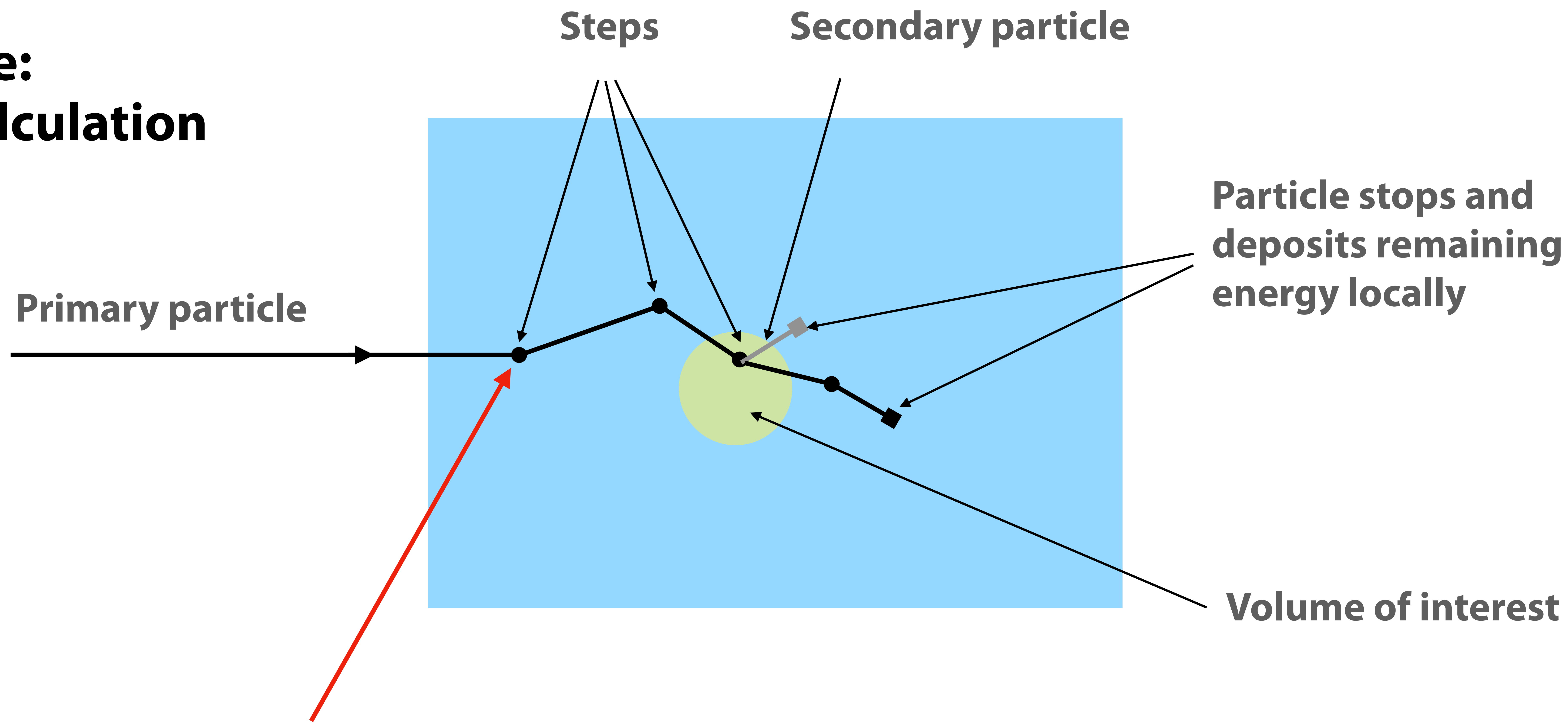
Application/physics specific:

- EGS (Electron, photon)
- Penelope (electron, photon)
- MCsquare (proton therapy)
- FRED (mainly fast dose calculation)
- GGEMS
- …

Particle transport simulation

Step-wise propagation of a particle across a medium

Example:
Dose calculation

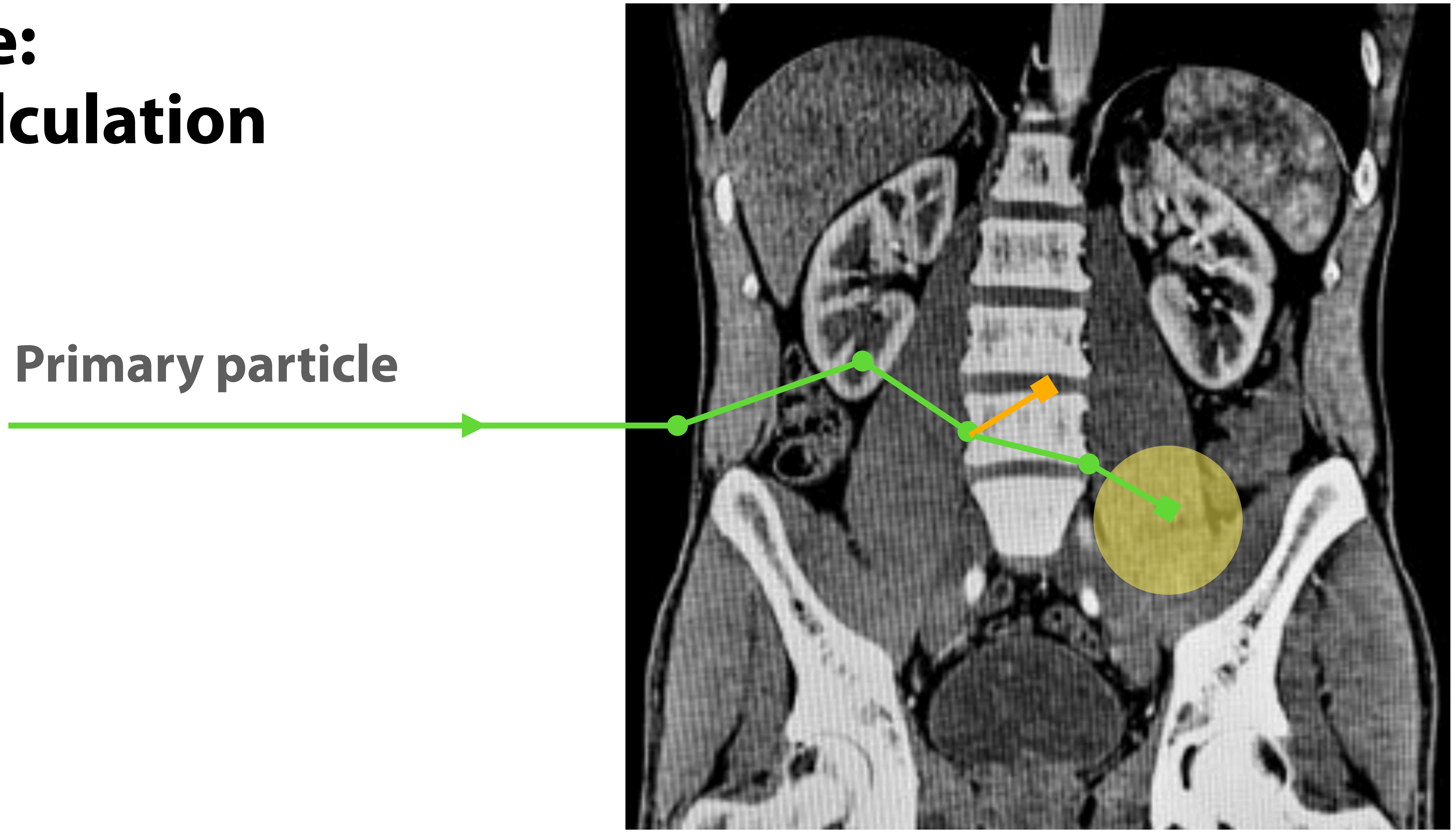


At each step: Evaluate and apply interaction models

Particle transport simulation

Patient = complex heterogeneous geometry

Example:
Dose calculation



Patient geometry usually parametrized via 3D discretized image: x-ray CT image

Ingredients of a Monte Carlo Simulation

Source (gammas, ions, …)

Geometry (objects, beamline, patient …

Physics (interactions of particles with the target, nuclear decay)

Output information about physics (dose, particle distribution, detector signal)

Ingredients: Closer look

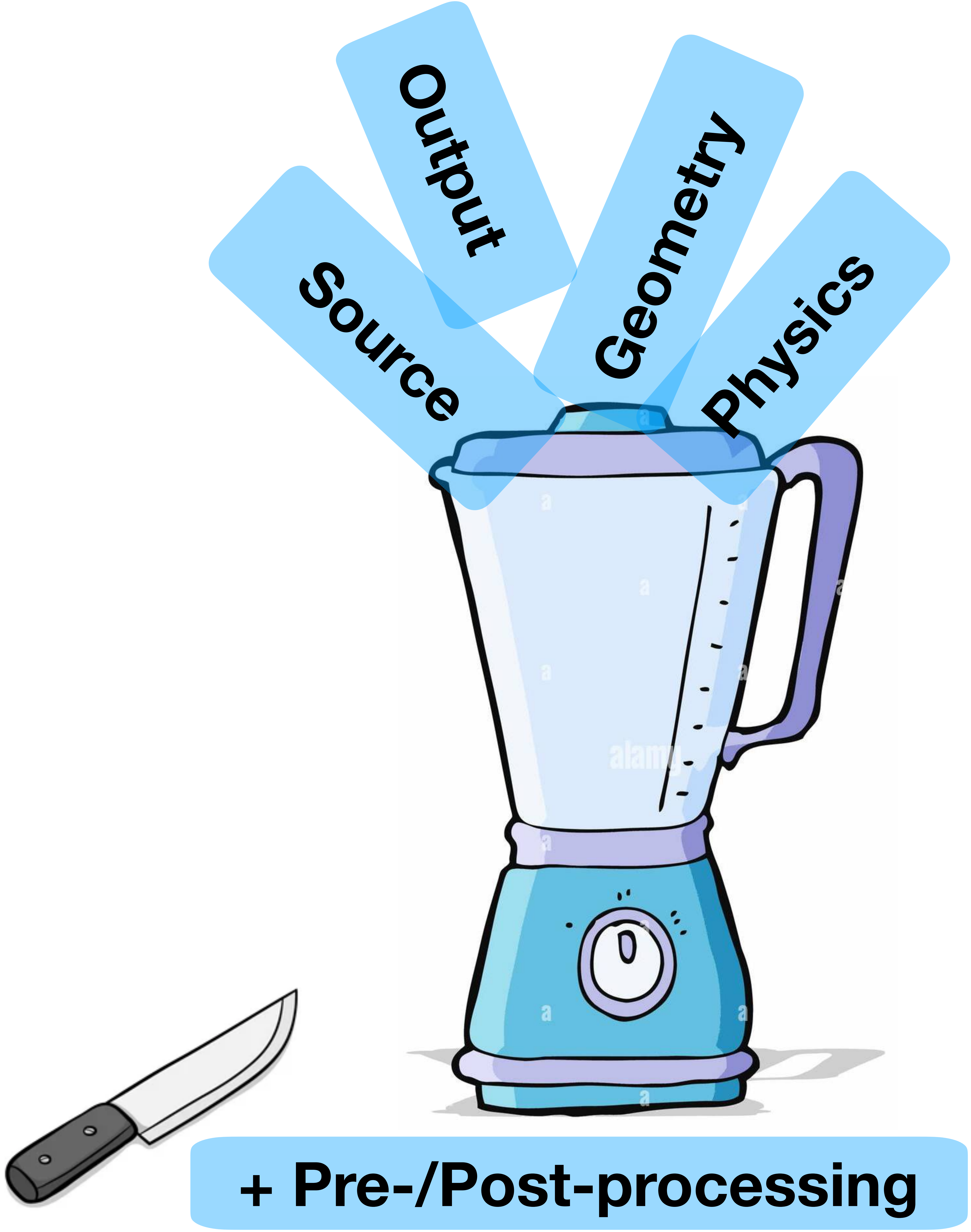
Output information about physics → **“Actors” in GATE**

Examples:

- Accumulate dose deposited inside a small cylinder inside a water box
- Record position and direction of all particles crossing a plane
- Record light output of a scintillator
  … and apply post-processing chain
- Record all prompt gammas generated by a proton beam

Mechanism:

Hook into the step-wise particle transport



GATE is the blender to mix the ingredients

Geant4 is the motor that makes the blender turn

How does GATE 10 work?

```
import opengate as gate

sim = gate.Simulation()

cm = gate.g4_units.cm
mm = gate.g4_units.mm
MeV = gate.g4_units.MeV

waterbox = sim.add_volume("Box", "Waterbox")
waterbox.size = [40 * cm, 40 * cm, 40 * cm]
waterbox.translation = [0 * cm, 0 * cm, 25 * cm]
waterbox.material = "G4_WATER"

source = sim.add_source("GenericSource", "Default")
source.particle = "proton"
source.energy.mono = 150 * MeV
source.position.radius = 10 * mm
source.direction.type = "momentum"
source.direction.momentum = [0, 0, 1]
source.n = 20000

dose = sim.add_actor("DoseActor", "dose")
dose.attached_to = waterbox
dose.size = [200, 200, 200]
dose.spacing = [2 * mm, 2 * mm, 2 * mm]

sim.run()
```

Write a few lines in python for geometry, source, physics, output recording

Execute the python script

Done

Easy for users in our field

Can be run in interactive python terminal, e.g. jupyter notebook

GATE 10 under the hood

Folder g4_bindings	Geant4 binding from C++ to Python (expose functions, classes) ; pybind11
Folder opengate_lib	Core classes (running): source, scorers etc

Folder opengate	User UI (initialisation)

Courtesy of David Sarrut

GATE 10 under the hood

Setup simulation: Volumes, sources, actors, physics etc.

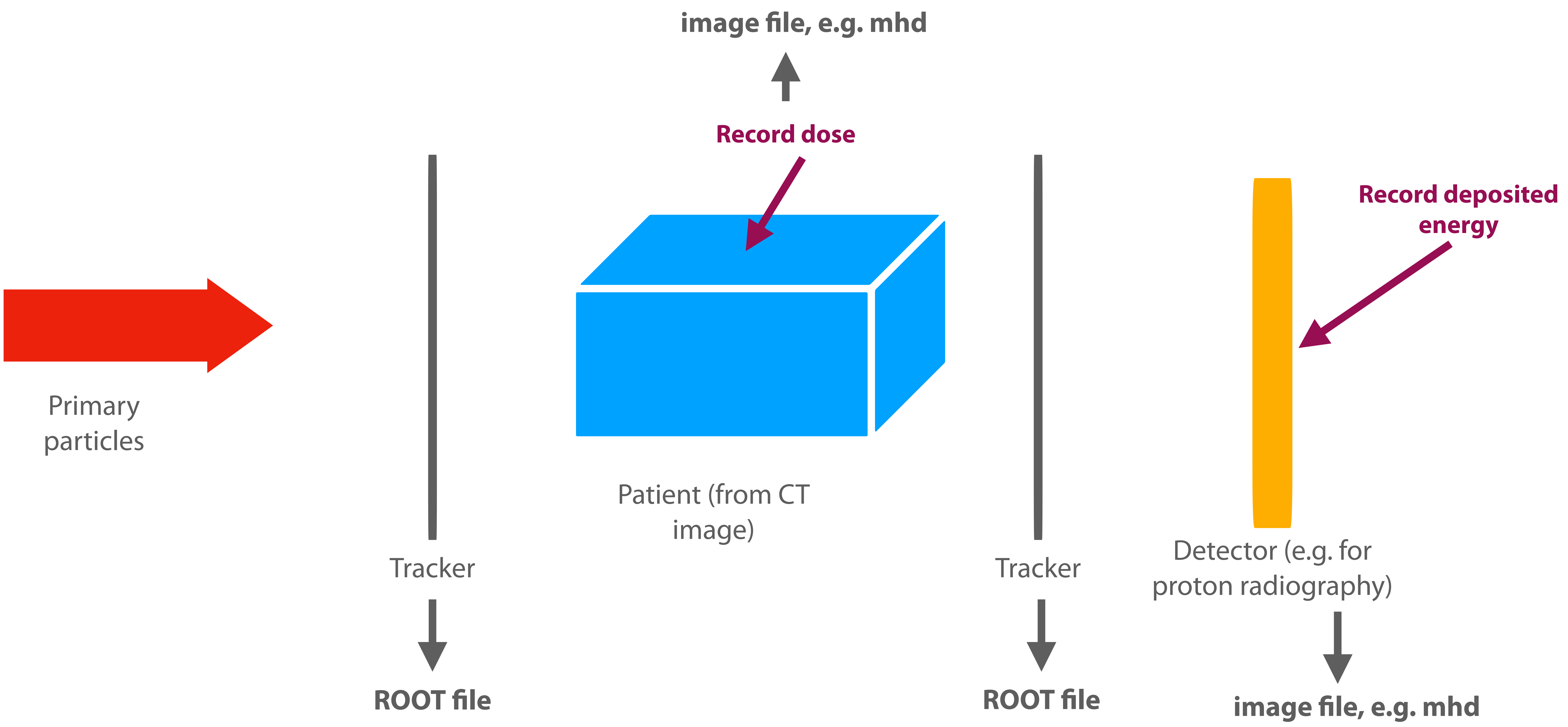
GATE 10 starts “engines”
and creates Geant4 objects via the library interface

Geant4 executes the simulation via the G4RunManager

GATE 10 releases all G4 objects, destroys the
G4RunManager, and closes the engines

Output is available on python side

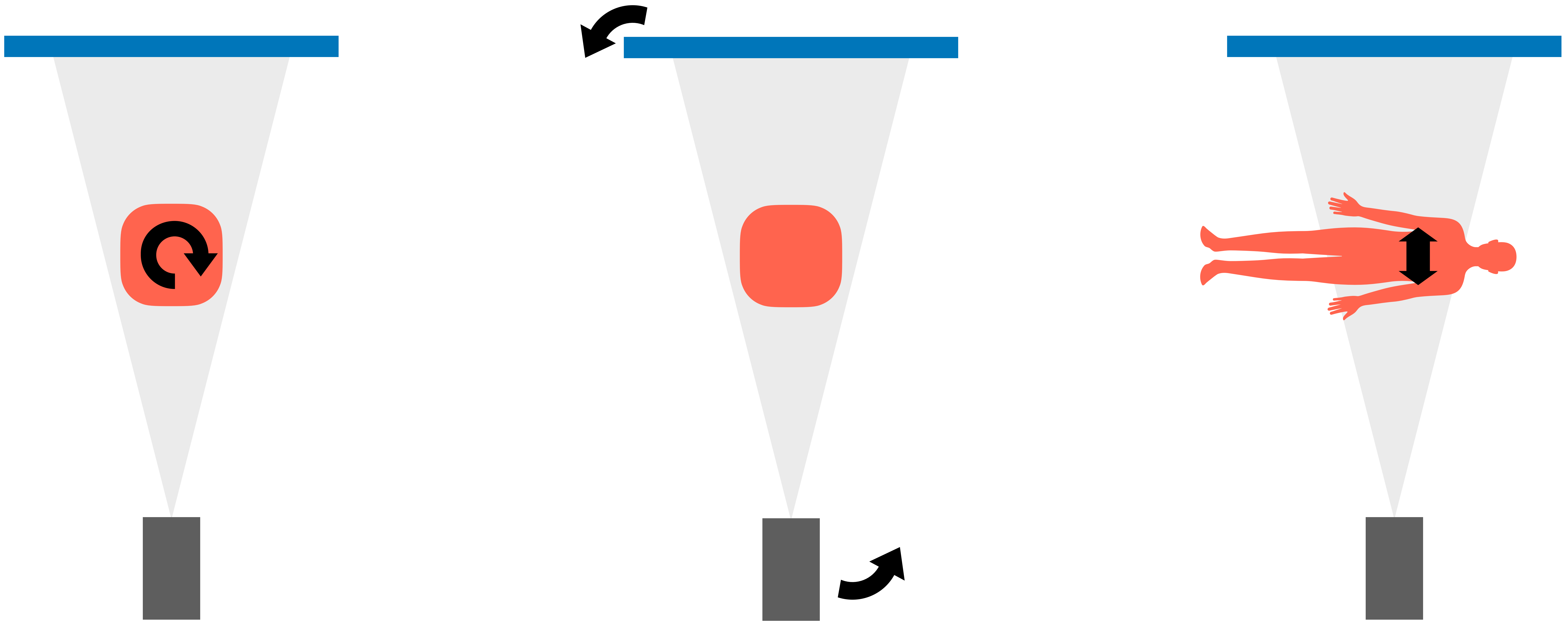
Example of “Monte Carlo Simulation” …



Let’s look at some code

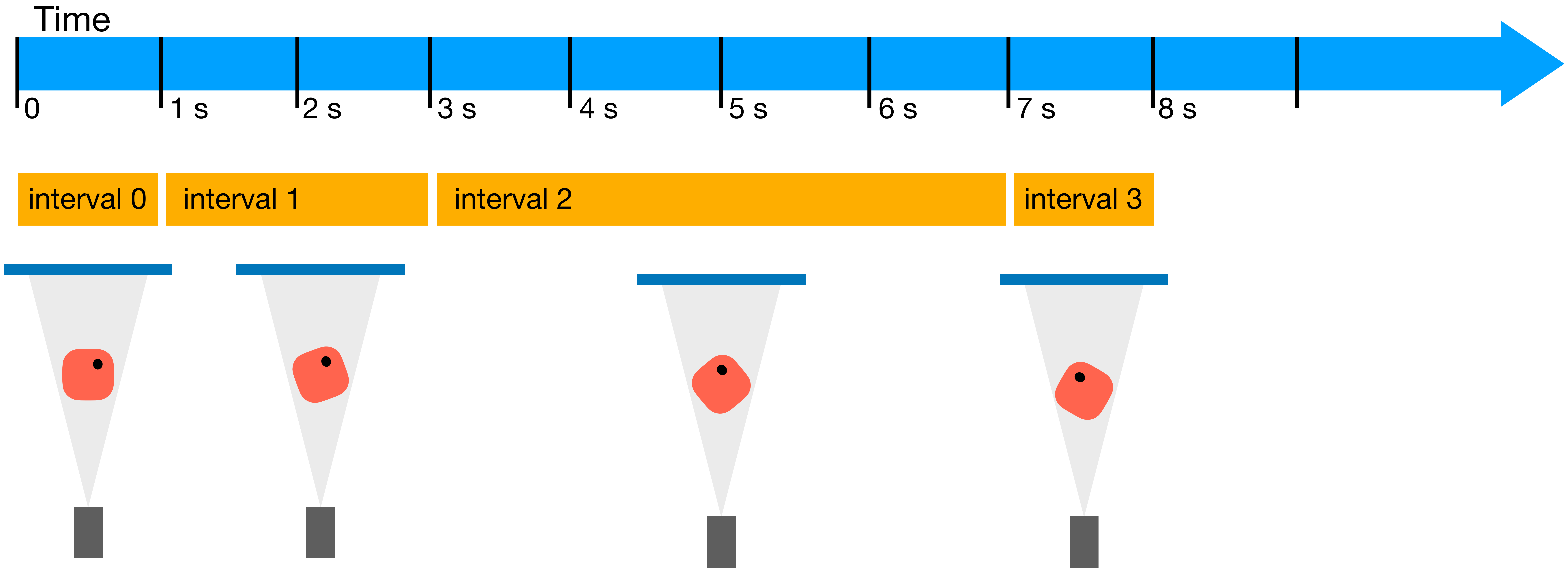
Dynamic parametrisations

- Typical examples: moving geometry, moving phantom (e.g. breathing patient)



Dynamic parametrisations

Simple code example: rotating target



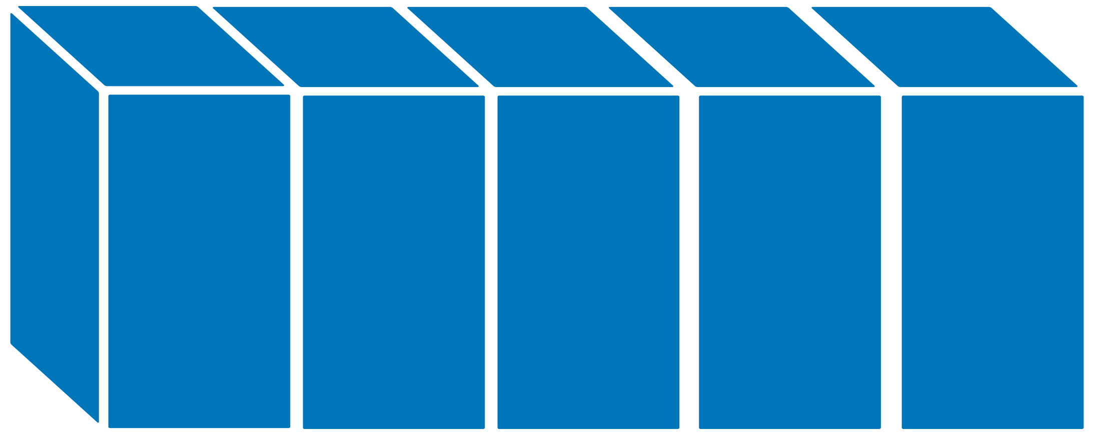
Let’s go back to the code

Repeated volumes

Repeating identical volumes is very simple in GATE 10.

Example: Scintillator strips in an ion CT tracker

Construct a 1D array of box volumes:



```
strips = sim.add_volume('Box', name='strips')

strip_width_in_mm = 0.5
mm = gate.g4_units.mm
strips.size = [strip_width_in_mm * mm, 3 * mm, 100 * mm]

strips.translation =
     [[t * mm, 0, 0] for t in np.arange(-20, 20, strip_width_in_mm)]
```

GATE creates multiple Geant4 Physical Volumes for the same GATE volume.

Ion CT Reconstruction pipeline

Single tracking = list-mode operation; pseudo-realistic

1 ROOT file per tracker → Add uncertainties → Tomographic reconstruction

Single tracking = list-mode operation; realistic

1 ROOT file per strip → Combine into 1 file per tracker → Tomographic reconstruction

---

Integrated mode operation, e.g. with a flat panel

Sinogram directly available → Tomographic reconstruction

Software for
CT reconstruction:
RTK (PCT),
Iterative codes

Store simulation as JSON

Here is what you can do in GATE 10:

```
sim = gate.Simulation()
…
# set up the simulation
…
sim.store_json_archive = True
sim.json_archive_filename = “sim_with_super_strange_physics.json”

sim.run()
```

This will create a structured, human-readable text file in JSON format at the end of the simulation.

Caveat: Does not work for actors and sources so far.

Store simulation as JSON

Screenshot of example simulation JSON file:


```json
{
    "user_info": {"name": "simulation"...},
    "object_type": "Simulation",
    "object_type_full": "<class 'opengate.managers.Simulation'>",
    "class_module": "opengate.managers",
    "i_am_a_gate_object": true,
    "volume_manager": {
        "user_info": {"name": "VolumeManager"...},
        "object_type": "VolumeManager",
        "object_type_full": "<class 'opengate.managers.VolumeManager'>",
        "class_module": "opengate.managers",
        "i_am_a_gate_object": true,
        "volumes": {
            "world": {"object_type": "BoxVolume"...},
            "rod": {"object_type": "TubsVolume"...},
            "waterbox_with_hole": {
                "user_info": {"name": "waterbox_with_hole"...},
                "object_type": "BooleanVolume",
                "object_type_full": "<class 'opengate.geometry.volumes.BooleanVolume'>",
                "class_module": "opengate.geometry.volumes",
                "i_am_a_gate_object": true
            },
            "patient": {"object_type": "ImageVolume"...}
        },
        "parallel_world_volumes": []
    },
    "physics_manager": {"object_type": "PhysicsManager"...}
}
```

Contribute to GATE 10

GATE 10 is an open-source community project, just as GATE 9.x has been. Any contribution is welcome!

https://github.com/OpenGATE/opengate

More than 100 tests/examples in the repository to get you started.

Documentation (in progress). You can edit the doc online.

https://opengate-python.readthedocs.io/

Work with GATE 10 as a user

- Install GATE 10 on your machine and start working with it.
- **pip install --pre opengate**
- Ask questions via the GATE mailing list:
  see http://www.opengatecollaboration.org
- Report issues via github:
  https://github.com/OpenGATE/opengate/issues

Every feedback is welcome!

Thanks for listening