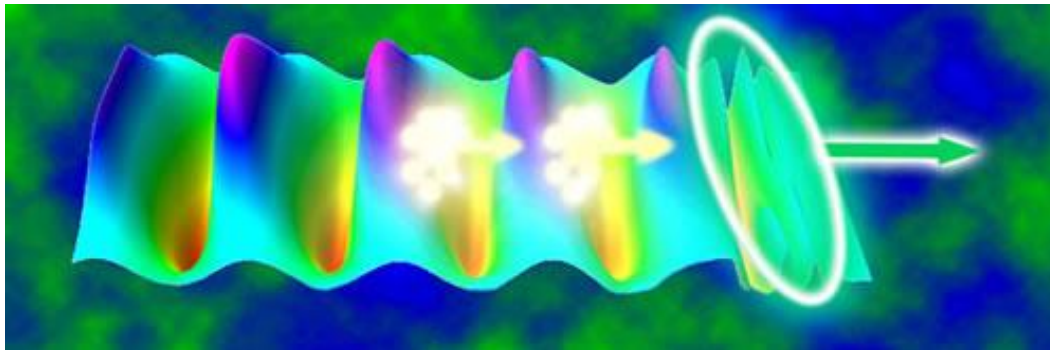




Dr. Alexei Sytov
INFN Ferrara Division

Use of geant4 in a laser-plasma environment

One-Week School on the Geant4 simulation toolkit

ELI 31/07/2026

World of particle accelerators

CERN
Large Hadron Collider:
Construction
\$5 billion (collider only)
Operation: \$1 billion/year



Germany
DESY 4th gen
Synchrotron Light
Source PETRA-IV:
€1.54 billion



Pohang 4th gen X-FEL:
\$390 million



World of particle accelerators: European X-FEL, DESY

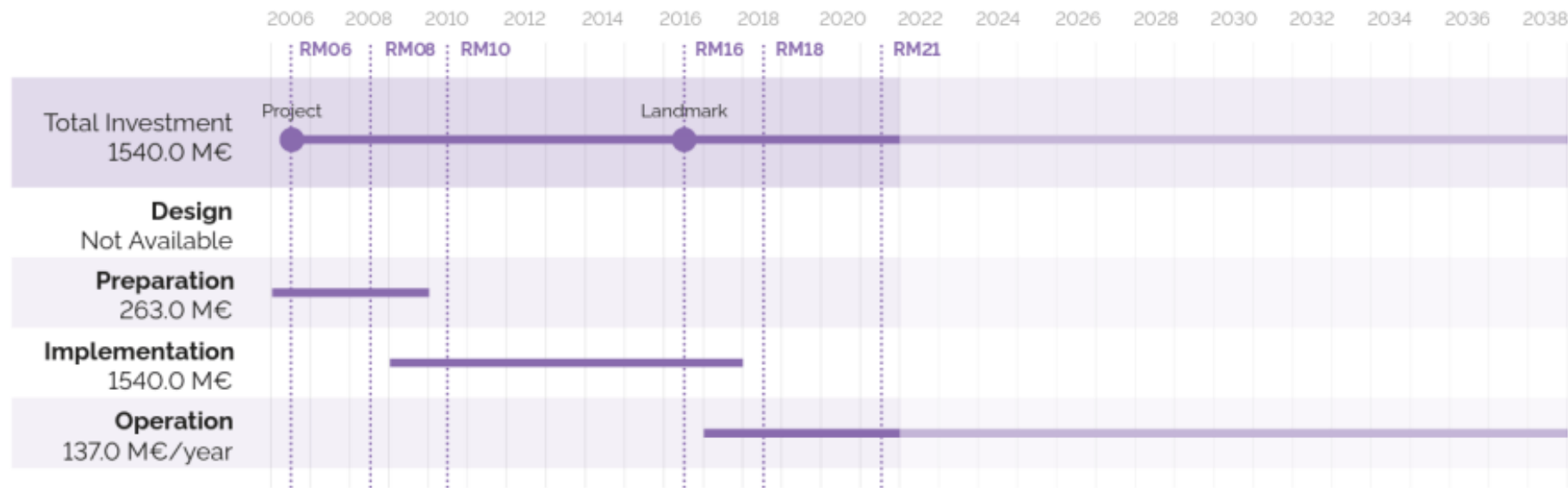


A huge «micro»scope

Recording chemical reactions on video using X-Rays

To perform R&D in

- Material science
- Nanotechnologies
- Genetics
- Viruses and vaccines



World of particle accelerators: cancer radiotherapy

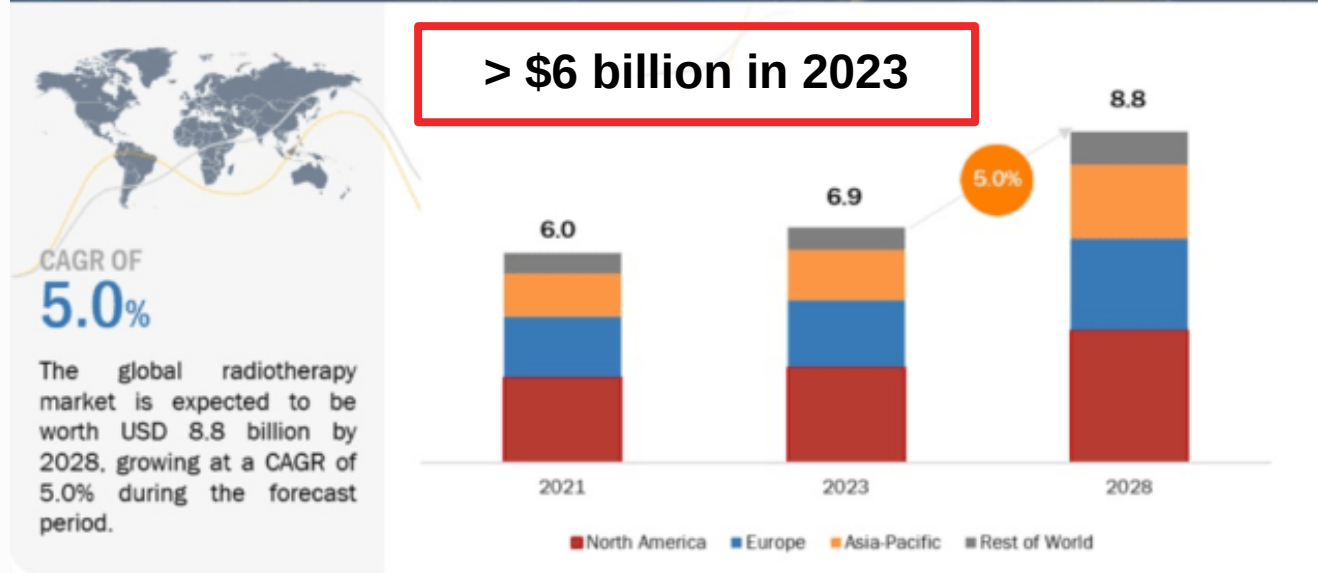


**7 million patients in
2023**

**5-year relative
survival rate > 90 %**

**Procedure cost
\$ 5000-50000**

RADIOTHERAPY MARKET GLOBAL FORECAST TO 2028 (USD BN)



**Cyclotron facility cost \$10 million
X-ray facility cost \$3 million**

World market of particle accelerators

> \$3 billion in 2023

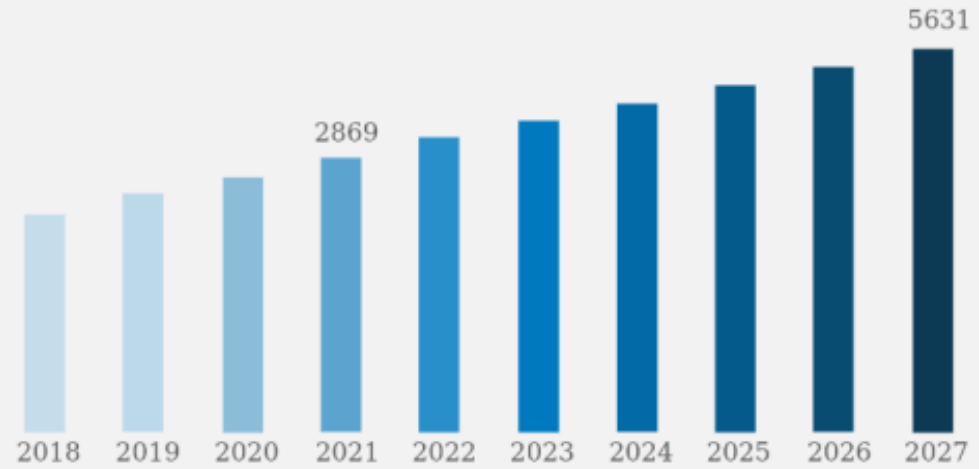
Global Particle accelerators market Share By Application,



- Healthcare
- Scientific Research
- Industrial

www.businessresearchinsights.com

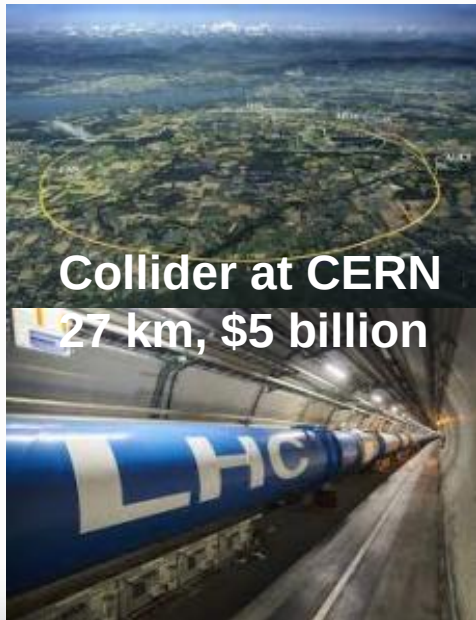
Global Particle accelerators market Size, 2027 (USD Million)



www.businessresearchinsights.com

Idea of plasma wakefield acceleration

Particle accelerators
are huge and
dramatically
expensive!



Prof. Toshiki Tajima



Prof. John W. Dawson

Idea:
to shrink the
accelerators to
the table-top size

We can use
plasma to
accelerate
charge particles

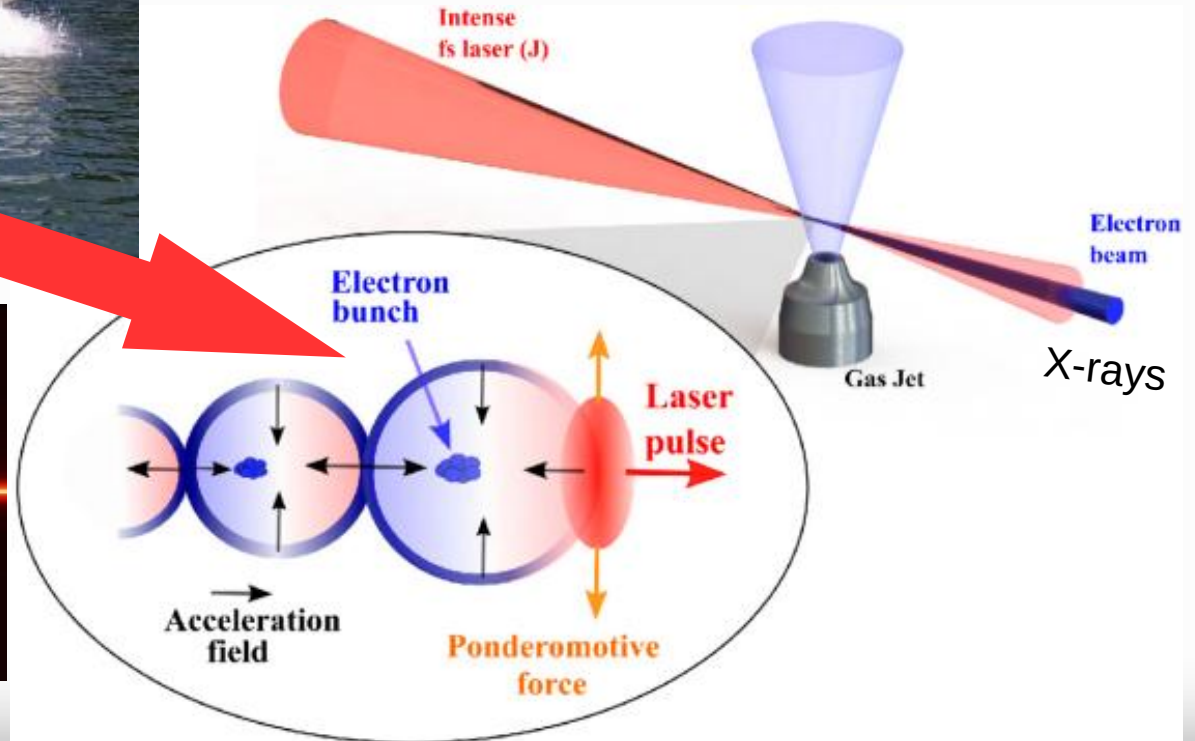
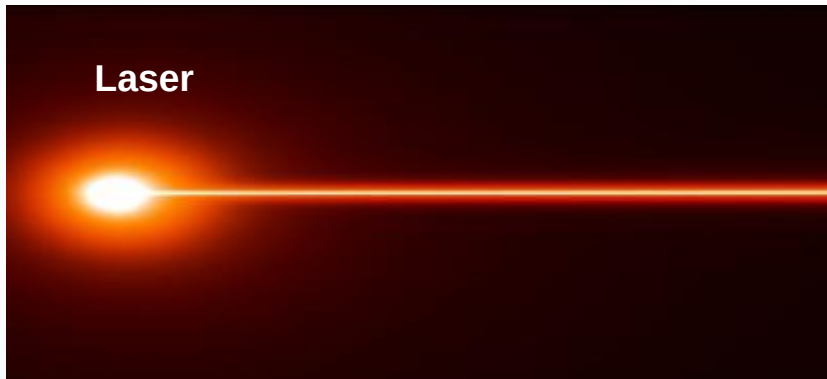
But:
we need a very
powerful laser



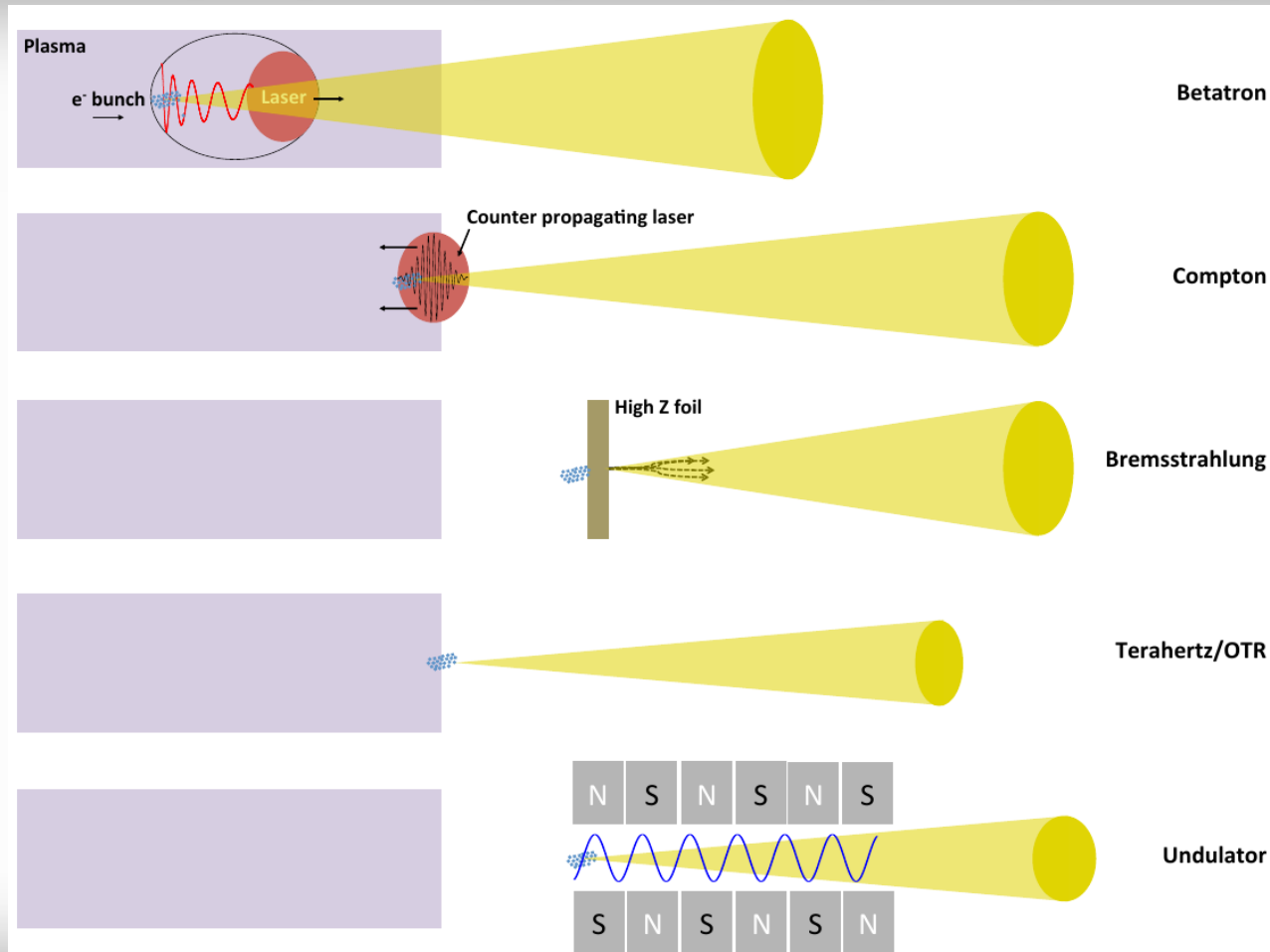
Physics of plasma acceleration: just like wakesurfing!



Electron beam energy: MeV — tens of **GeV**



Radiation source based on plasma acceleration of electrons

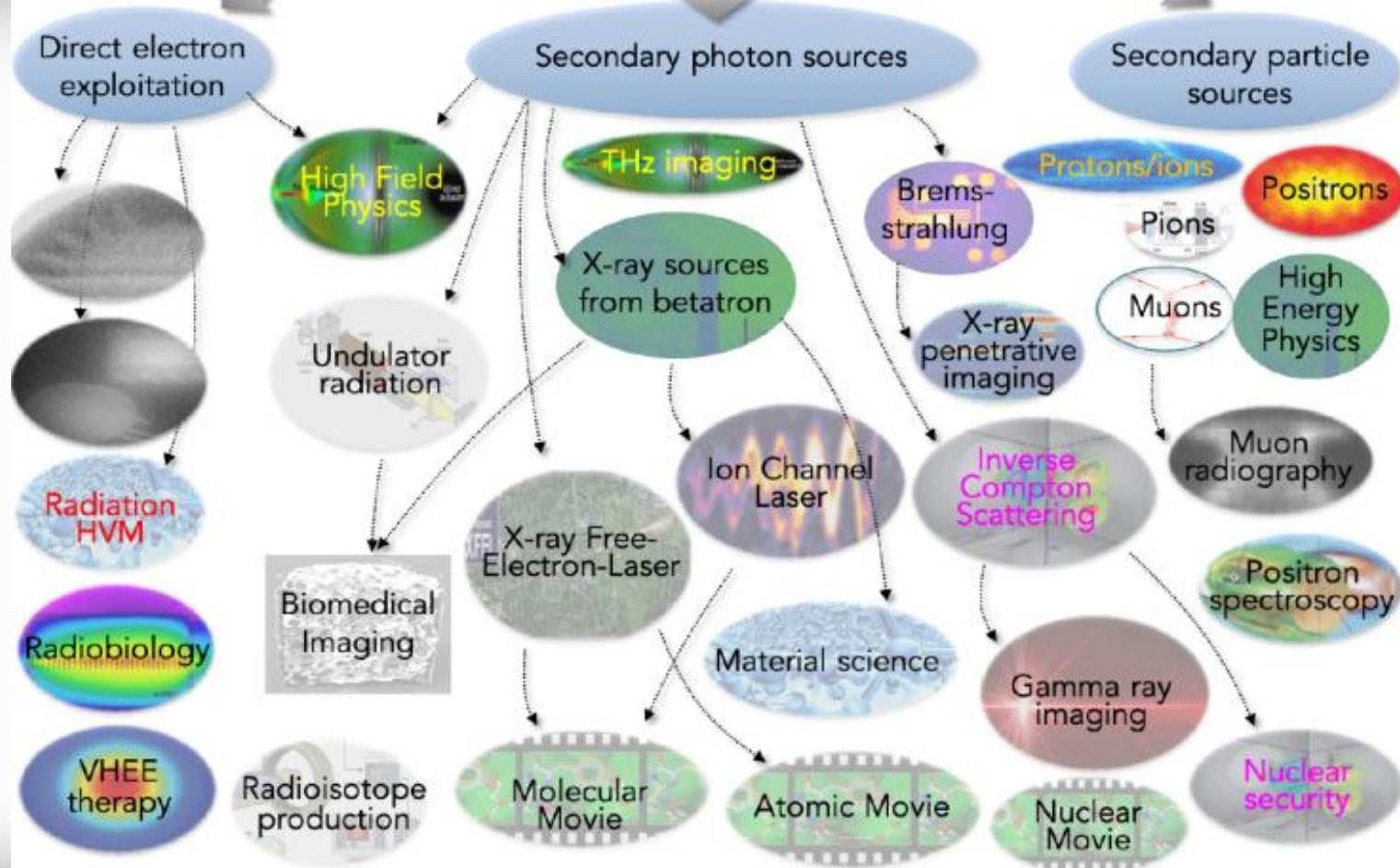


X-rays

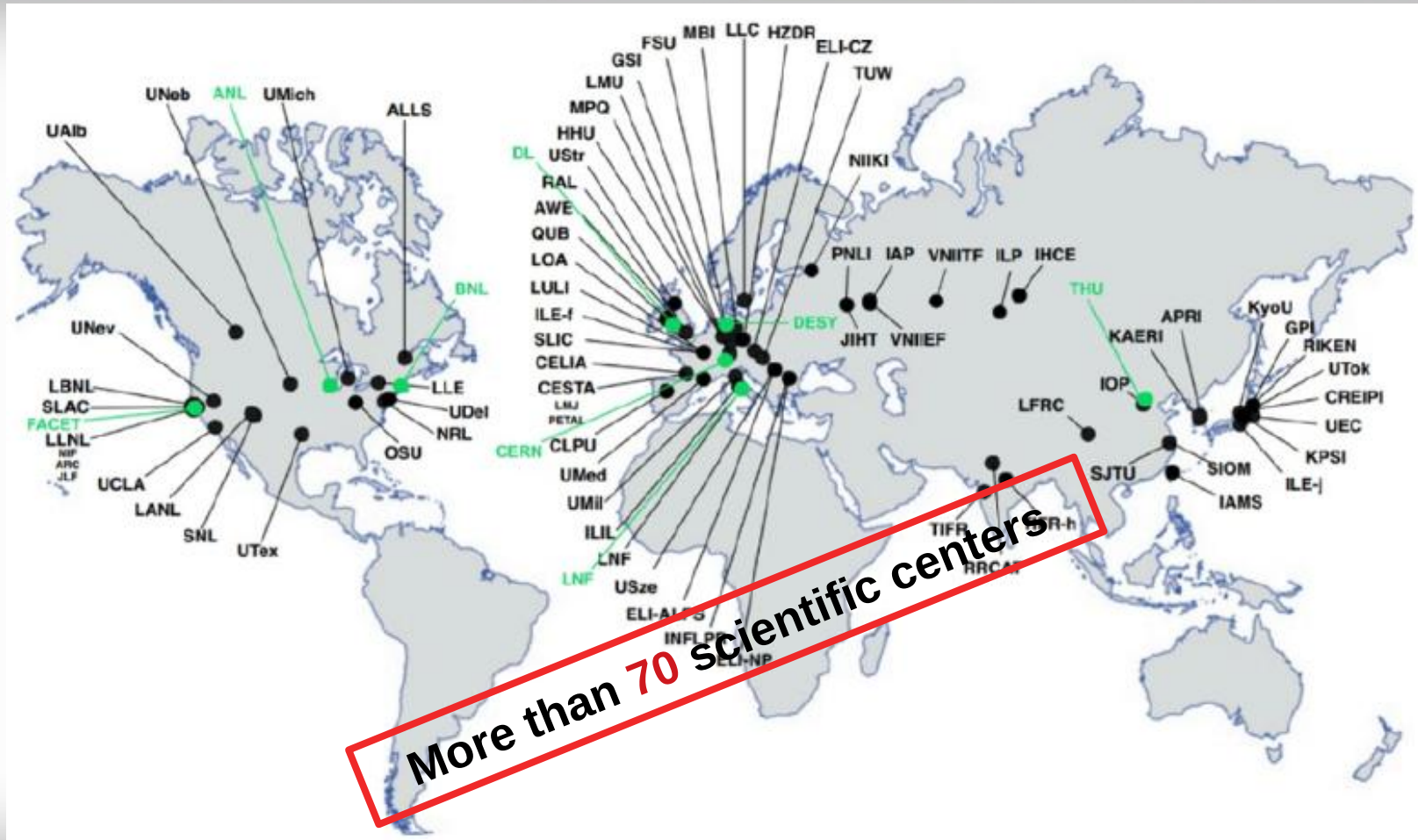
γ-rays

Wakefield Accelerated Electrons

LWFA; e-PWFA; L→PWFA; p-PWFA



Scientific centers working on plasma wakefield acceleration



Example of a big project in plasma acceleration: EuPRAXIA

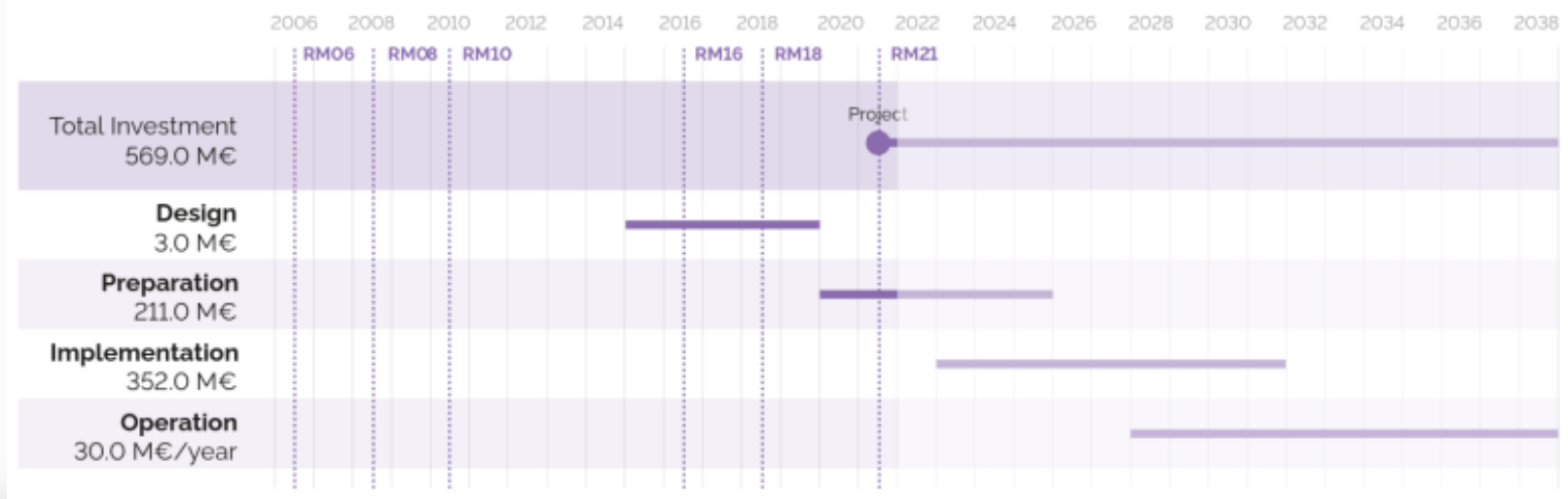


December 2020,
EuPRAXIA consortium:
40 members; 11 observers.

Hosted by INFN, Italy



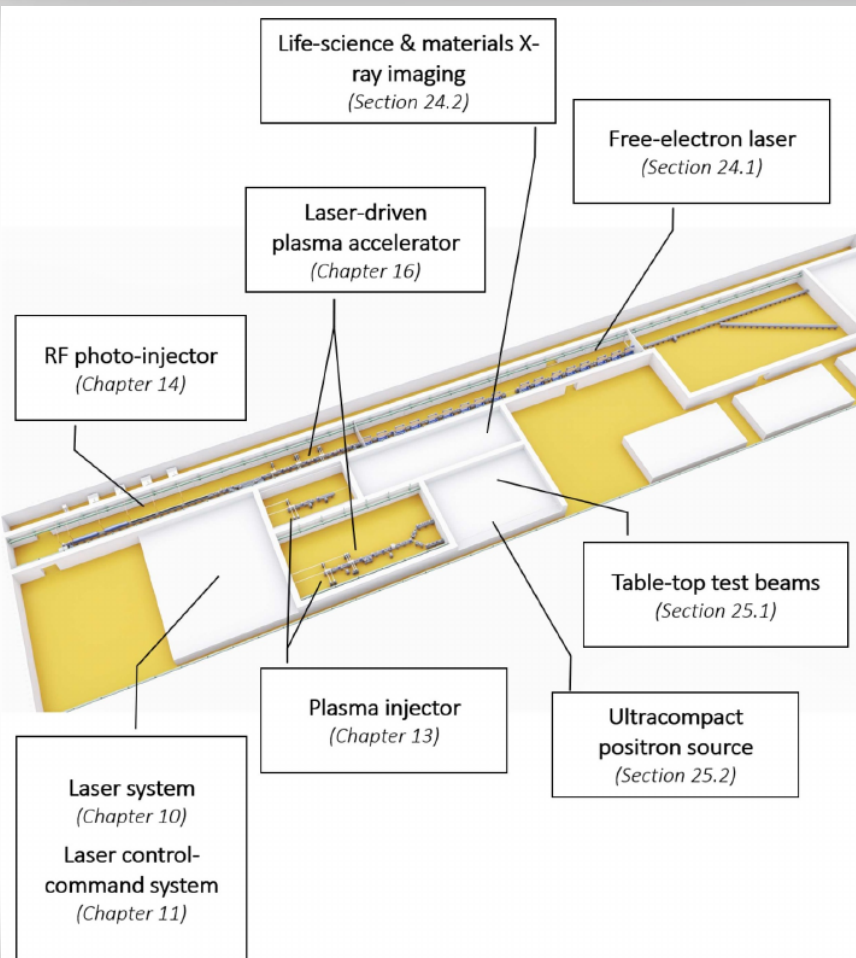
Operation
until 2062



Laser-driven plasma acceleration @EuPRAXIA

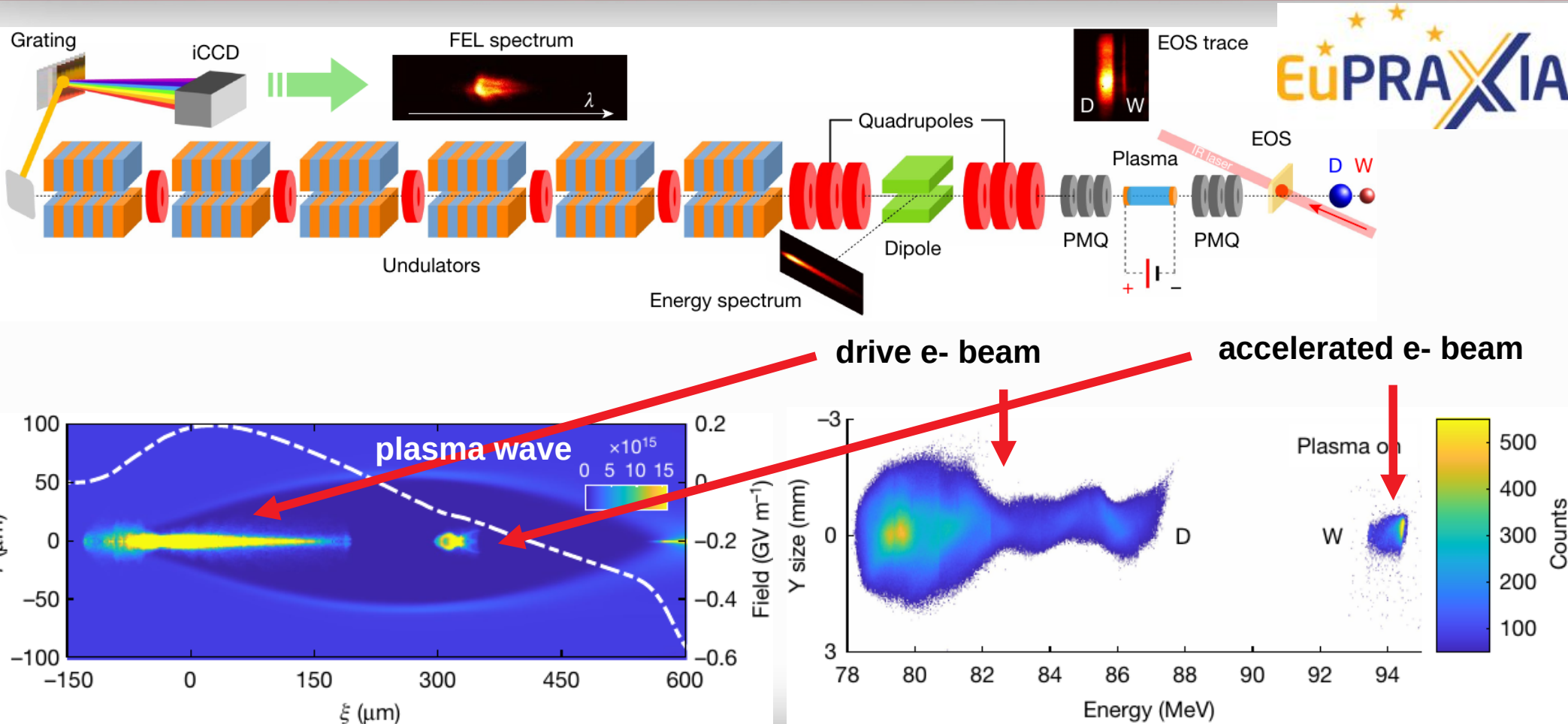


Betatron radiation facility



Quantity	Baseline Goal	Range of Exploration
Energy	5 GeV	≥ 1 GeV
Charge	30 pC	5–100 pC
Bunch length (FWHM)	10 fs	≤ 30 fs
Peak current	3 kA	2–5 kA
Energy spread	1 %	< 1 %
Normalised emittance in x,y	1 mm mrad	< 1 mm mrad
Slice energy spread	0.1 %	< 0.1 %
Normalised slice emittance in x,y	< 1 mm mrad	$\ll 1$ mm mrad

Beam-driven plasma acceleration @EuPRAXIA



Examples of investment in plasma acceleration: ELI



lead country
CZ, HU, RO
members countries
DE, FR, IT, UK

ELI

Extreme Light Infrastructure

The world's fastest and most intense advanced lasers to unravel the fundamental secrets of the Universe and materials



TIMELINE

Roadmap Entry
2006

Preparation Phase
2007-2010

Implementation/Construction Phase
2011-2017

Operation Start
2018

ESTIMATED COSTS

capital value
850 M€

design
Not Available

preparation
6 M€

construction
850 M€

operation
80 M€/year

**This is where the most
power lasers are!**

**This is where
plasma borns!**

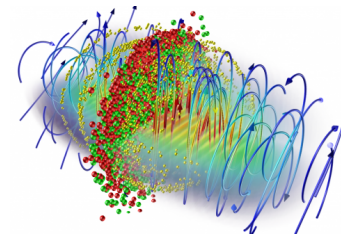
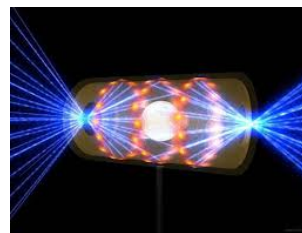
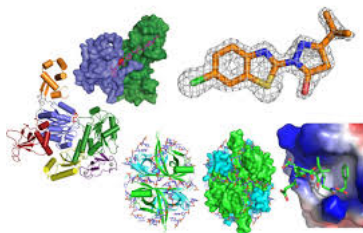
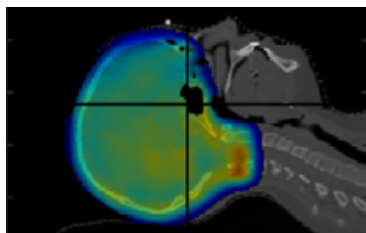


(User) Scientific Mission of ELI Beamlines



User Science Areas and Mission

- ✓ Exploit **high-energy**, high-flux, high-repetition-rate (**high average power**) **particles and photons** through dedicated beamlines driven by PW-class lasers (ultrafast **radiobiology**, ultrahigh dose-rate/FLASH **radiotherapy**, and **material science**)
- ✓ Deliver **ultrahigh laser intensities** ($>10^{22} \text{ W/cm}^2$) on target for **high-field physics** (kJ-class and 10s-J-class, high rep. rate lasers), including **non-linear QED**
- ✓ Multi-beams in exp. stations for **pump-probe** setups to study **ultrafast dynamics** in the **kHz** regime (**AMO science** and **structural biology**) and in the **kJ** regime (**HEDP**, including **IFE** research)

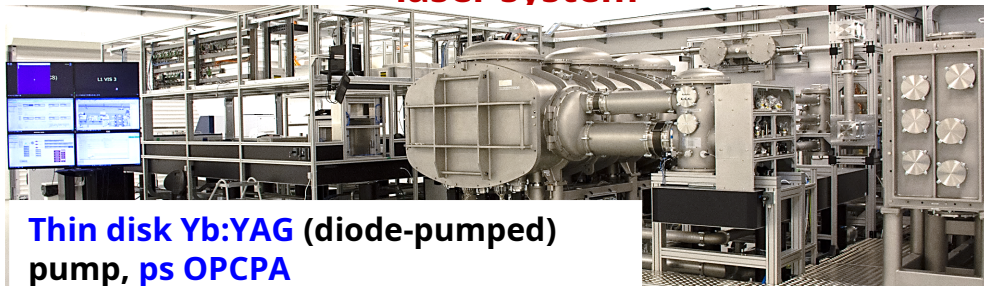




ELI BL Primary Sources (Lasers)

high-energy, high-rep-rate laser systems (0.01-1 kHz)

L1-ALLEGRA 100mJ/12fs/1kHz (>5TW)
laser system



Thin disk Yb:YAG (diode-pumped)
pump, ps OPCPA

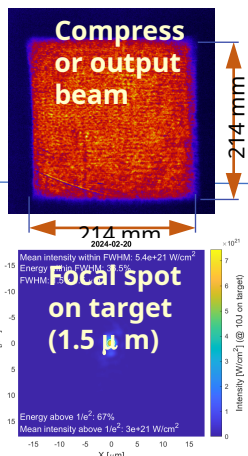
Auxiliary 15 mJ /12 fs beam (Q4 2024)

L3-HAPLS: 30J/30fs/10Hz (1PW) laser system
(in operation - 15J/27fs/3.3Hz)



He-cooled (diode-pumped)
Nd:glass pump, Ti:Sa

Ramping to 1PW level (Q3 2025)



Slide by Daniele Margarone

L2-DUHA: 3J/20fs/50-100Hz (>100TW) laser system
(under development)



Cryogenic He-cooled slab Yb:YAG (diode-pumped) pump, ns OPCPA

Expected availability for experiments: Q4 2025

L4-ATON: 1.5kJ/150fs/1shot/min (10PW) laser system
(long pulse in operation - 1.2kJ (0.6kJ @2w)/2-10ns/1shot/min)

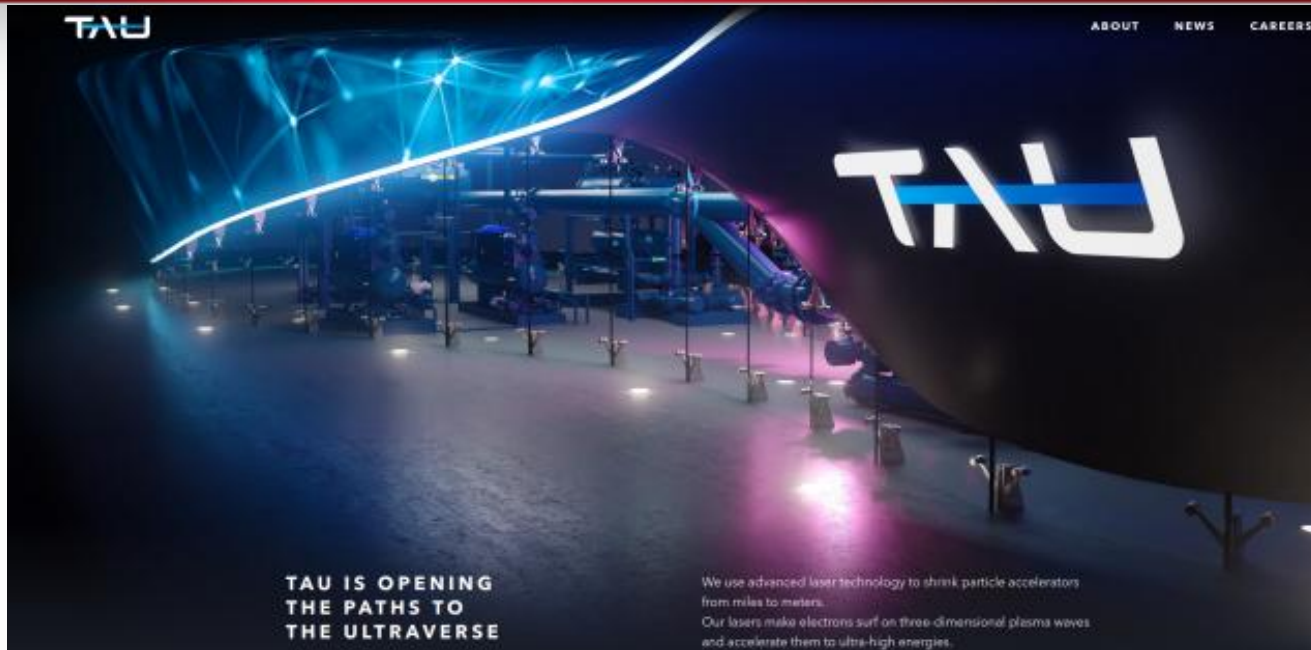


Mixed Nd:glass (>15nm spectral bandwidth)
ns-kJ pulses with programmable temporal
shaping (150ps steps)

10PW commissioning on target: Q3 2025



Startups in plasma acceleration are coming!



TAU Systems, US

Created in 2022

**Attracted \$15 million
from a business angel**

Claim to built first commercial X-FEL based on plasma acceleration by 2027

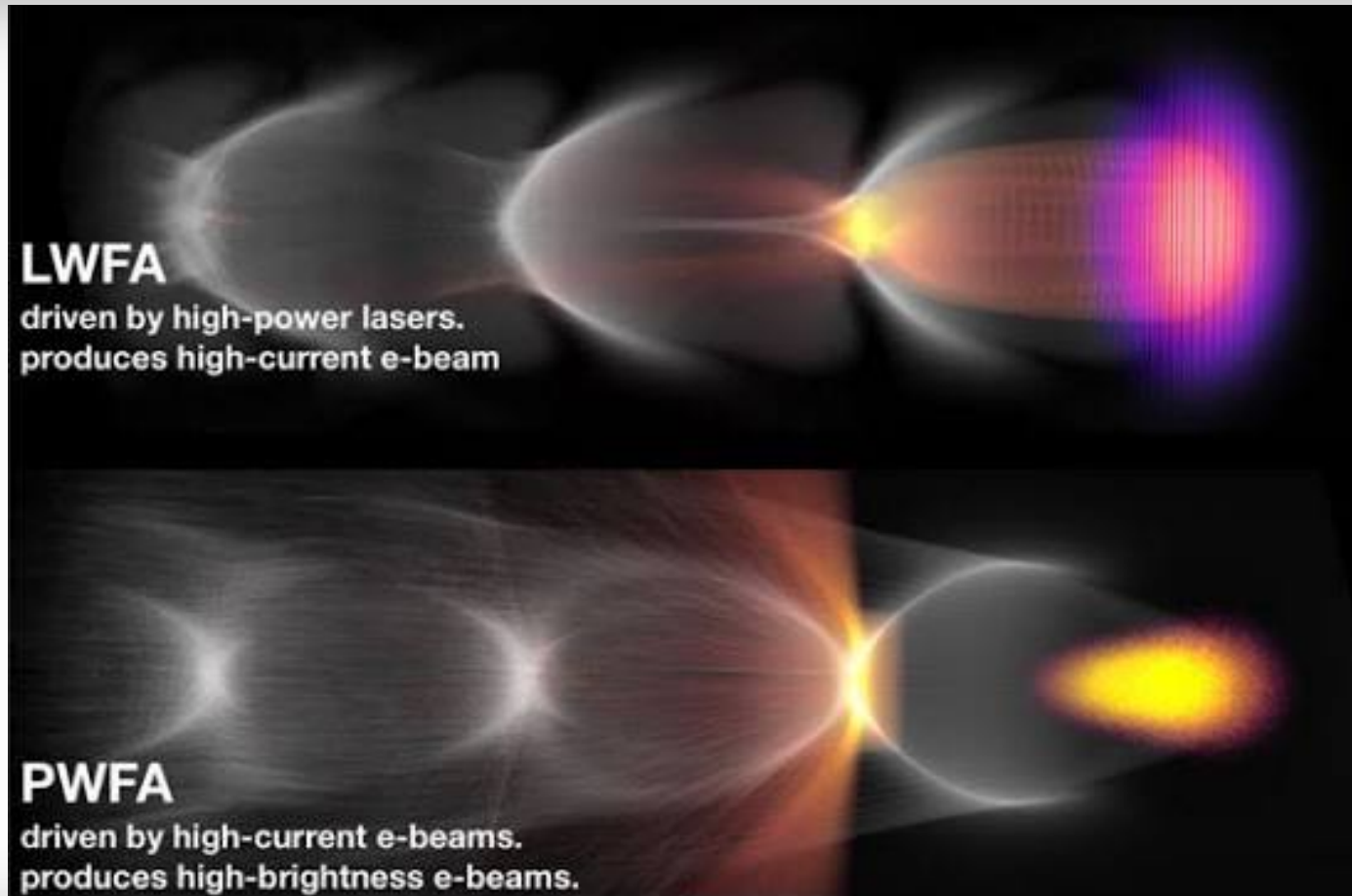
A probable laser facility for the first proof-of-concept experiment

Laser Power		350 TW
Energy per pulse		>7 J
Pulse duration		≤ 25 fs
Focusing surface		$36 \mu\text{m}^2$ or better
Max power density (at the target)		$8.82 \cdot 10^{20}$
$I \cdot \lambda^2$		$5.64 \cdot 10^{20}$
Contrast ratio @100 ps (ASE)		$> 10^{10}$
Repetition rate		1 Hz
Protons Ions	Max energy	50 MeV
	Particle per pulse (at 30 MeV)	$10^{11} \text{ MeV}^{-1} \text{ Sr}^{-1}$
	Energy spread	100%
	Beam divergency (max)	$\pm 20^\circ$
Eletrons	Max energy	3 GeV
	Particles per pulse	10^9
	Beam divergency (max)	$\pm 20 \text{ mrad}$
Neutrons	Max energy	20 MeV
	Particles per pulse	10^{10}
	Energy spread	100
	Beam divergency	Isotropic

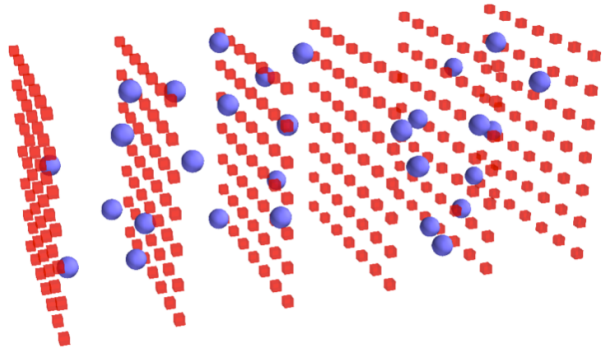


Slide by Pablo Cirrone

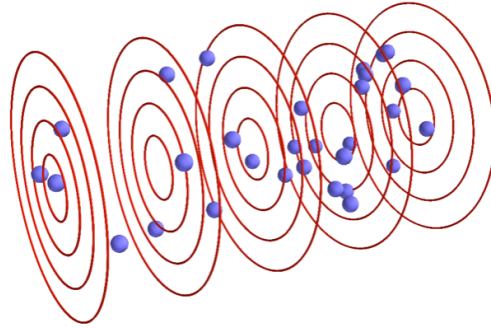
Example of Particle-in-cell (PIC) simulation



Particle-in-cell (PIC) simulation concept



3D Cartesian grid

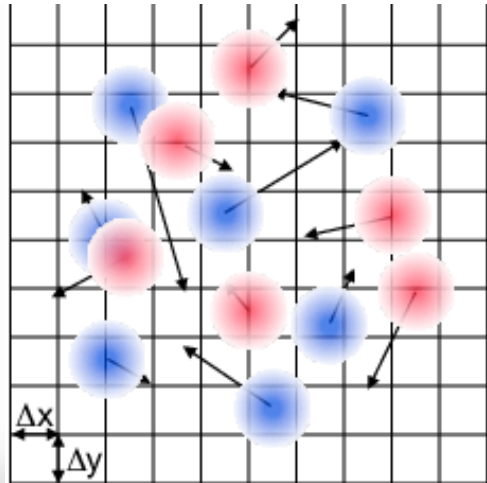


Cylindrical grid (schematic)

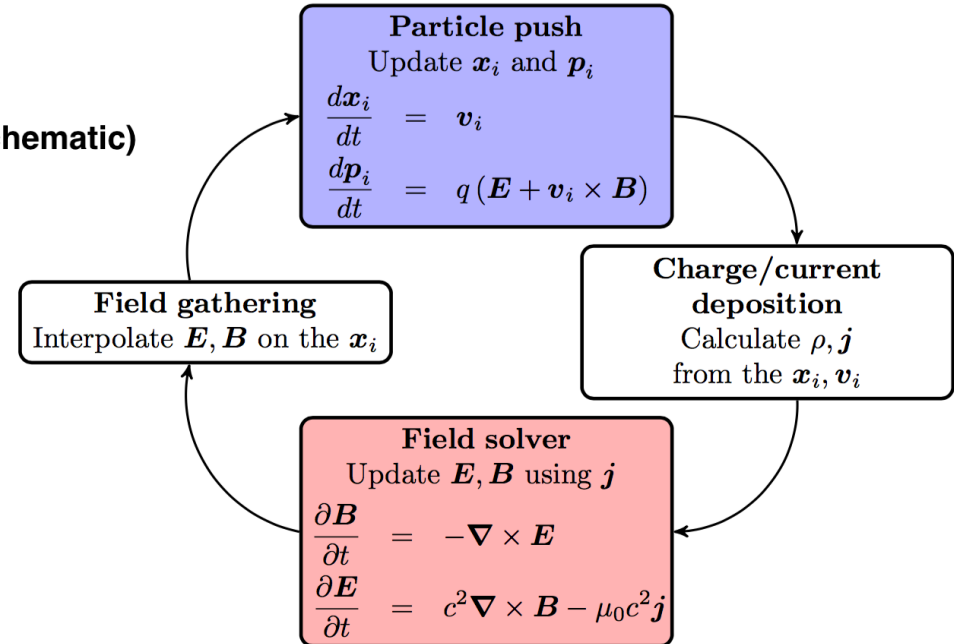
Meshing of space
into cells: 1D/2D/3D

Interpolation of
fields

Finite time steps



Particles => macroparticles
(with proportional charge
and mass)



Monte Carlo vs Particle-in-cell

Monte Carlo simulates independent events

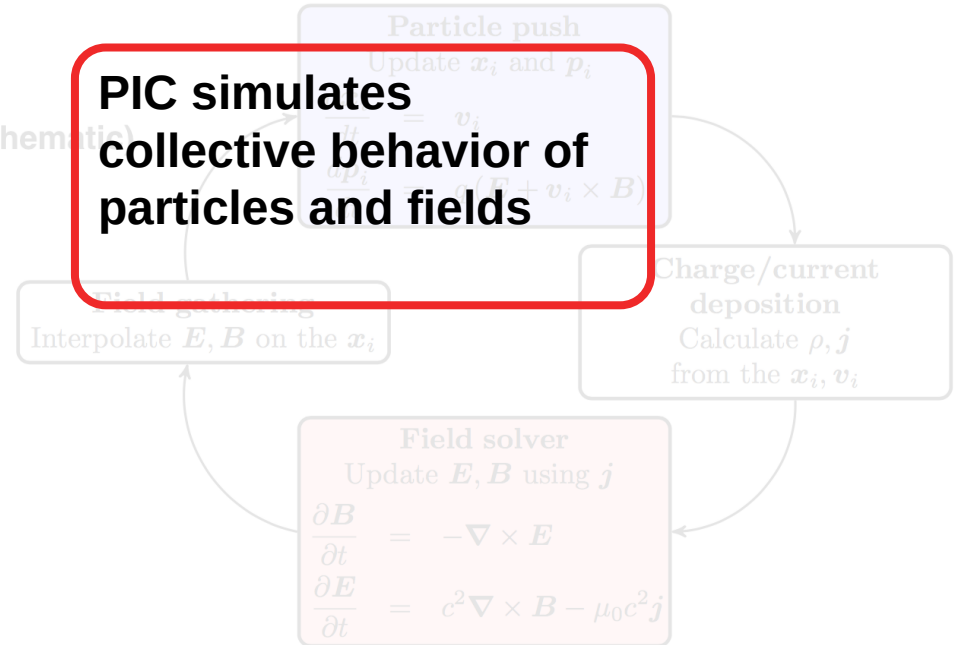
PIC simulates collective behavior of particles and fields

Particles => macroparticles
(with proportional charge and mass)

Meshing of space
into cells: 1D/2D/3D

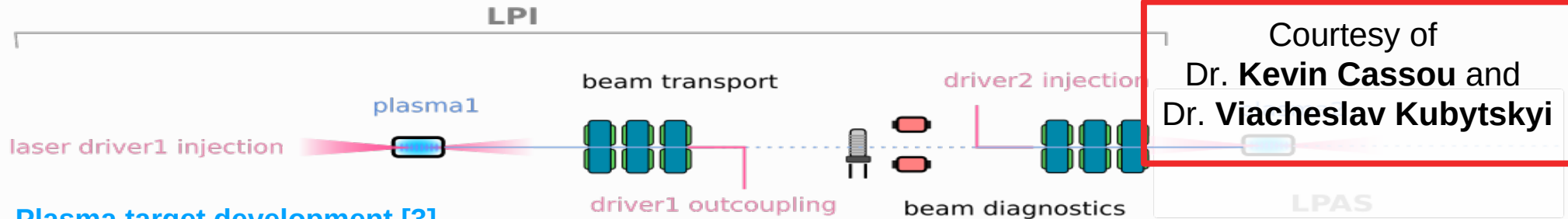
Interpolation of
fields

Finite time steps



Test facility for laser-plasma injector optimisation towards RF control reliability

In the context of advanced accelerator high quality beam laser plasma injector (LPI) for **EuPRAXIA** [1] preparatory technical design phase and future high gradient accelerator R&D at IJCLab [2]: 10 Hz 250MeV LPI test facility to improve **quality and stability of e- beam generated by laser-plasma accelerator**.



Plasma target development [3]

- testing various gas cell type
- Continuous flow
- In-lin integration

INFN Ferrara: Geant4 simulations&ML

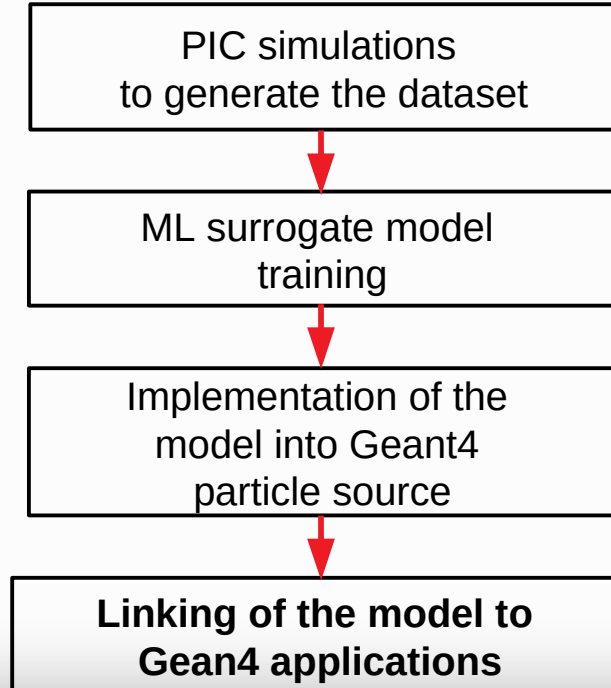
[1] : Assmann, R. W. et al. EuPRAXIA Conceptual Design Report. Eur. Phys. J. Spec. Top. 229, 3675–4284 (2020).

[2] pallas.ijclab.in2p3.fr

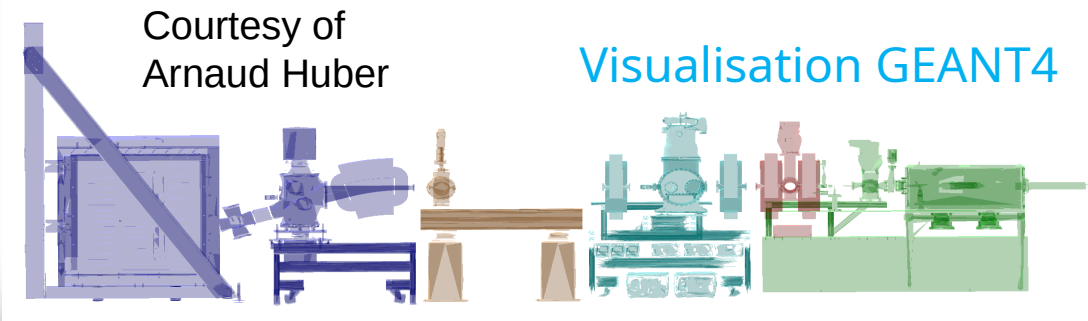
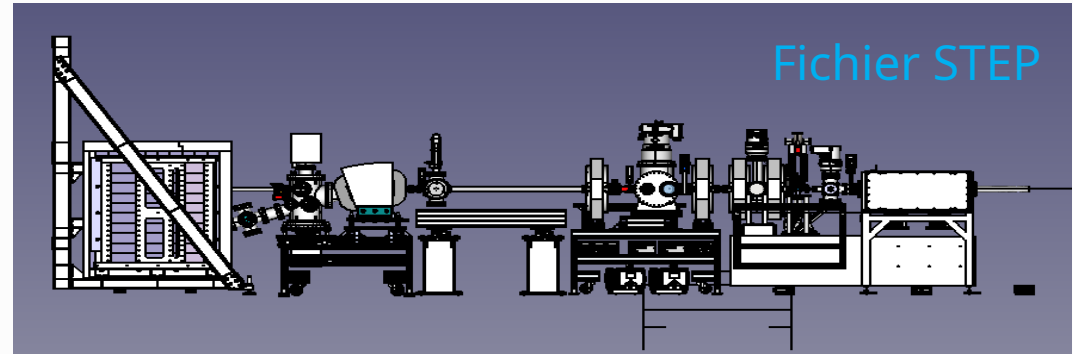
[3] Drobniak, P. et al. Random scan optimization of a laser-plasma electron injector based on fast particle-in-cell simulations. Phys. Rev. Accel. Beams 26, 091302 (2023), Drobniak, P. et al. Two-chamber gas target for laser-plasma accelerator electron source. Preprint at <http://arxiv.org/abs/2309.11921> (2023).

Option 1: ML Learning to implement plasma acceleration into Geant4

Implementation of ML surrogate model of laser-driven plasma wakefield acceleration trained with PIC simulations into Geant4



PALLAS Project: Test facility for laser-plasma injector Partner of EuPRAXIA



Option 1: ML Learning to implement plasma acceleration into Geant4

Implementation of **ML surrogate model of laser-driven plasma wakefield acceleration** into **PrimaryGeneratorAction**

```
#ifndef B1PrimaryGeneratorAction_h
#define B1PrimaryGeneratorAction_h 1

#include "G4VUserPrimaryGeneratorAction.hh"
#include "globals.hh"
#include "G4GeneralParticleSource.hh"
#include "G4ParticleGun.hh"
#include <memory>
#include <onnxruntime_c_api.h>
#include <onnxruntime_cxx_api.h>
class G4ParticleGun;
class G4Event;
```

- 1) Develop your neural network (you may use python).
- 2) Save your network into a file.
- 3) Upload it from Geant4. Use C++ frameworks: ONNX, pytorch, ...

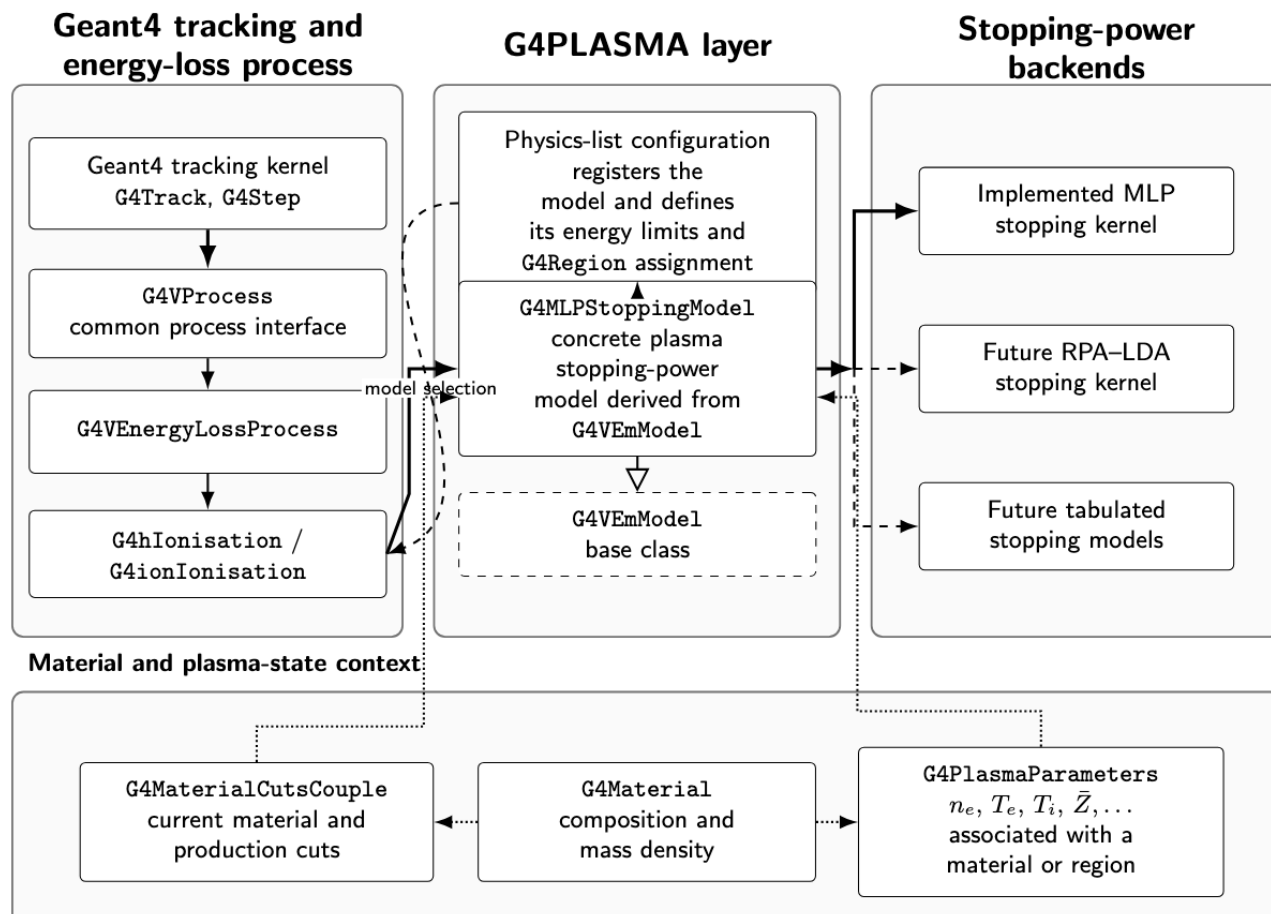
```
PrimaryGeneratorAction::PrimaryGeneratorAction(): G4VUserPrimaryGeneratorAction(),
    fParticleGun(0),
    fEnvelopeBox(0)
{
    G4int n_particle = 1;
    fParticleGun = new G4ParticleGun(n_particle);

    // default particle kinematic
    fParticleGun->SetParticleDefinition(
        G4ParticleTable::GetParticleTable()->FindParticle("e-"));

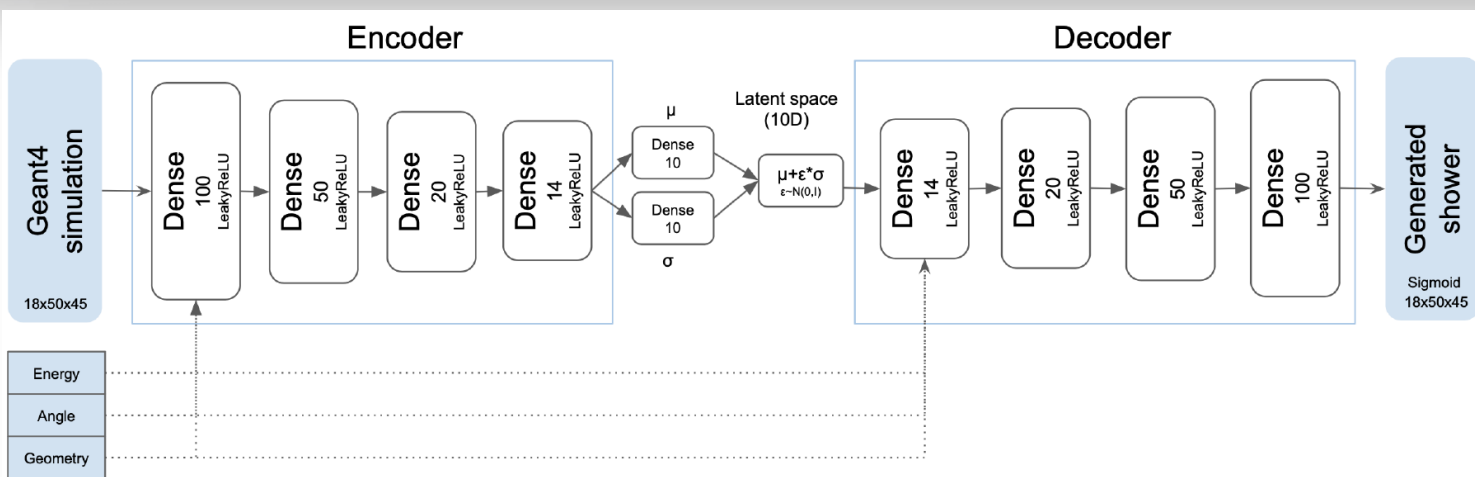
    //Neural network: create onnx session
    Ort::Env env(ORT_LOGGING_LEVEL_WARNING, "plasma");
    Ort::SessionOptions session_options;
    session_options.SetIntraOpNumThreads(1);
    auto sessionLocal =
        std::make_unique<Ort::Session>(env, "model2.onnx", session_options);
    fSession = std::move(sessionLocal);

    // Get input node information
    fMemory_info = Ort::MemoryInfo::CreateCpu(
        OrtAllocatorType::OrtArenaAllocator, OrtMemTypeDefault);
```

Option 2: Implementation of stopping power in plasma into Geant4 (by the INFN LNS group organizing this school)

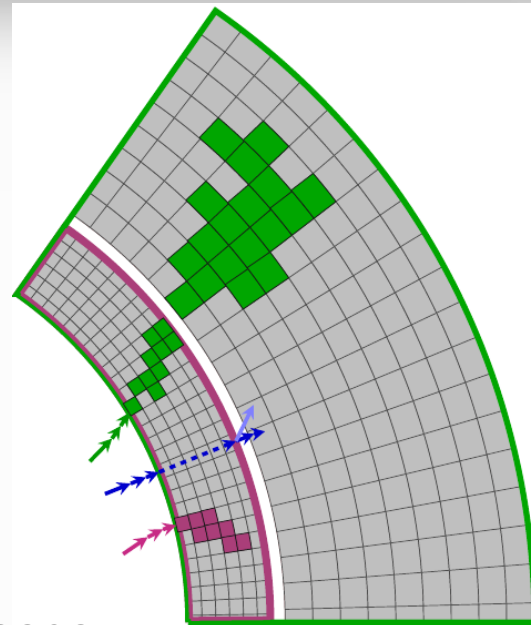


Option 3: Fast Simulation Interface



Easy way to implement any external code into Geant4

- **Fast simulation** completely stops the standard Geant4 processes at the step of Fast Simulation model and then resumes them
- Is activated **only** in a **certain G4Region** at a **certain condition** and only for **certain particles**



How to create your own Fast Simulation Model?

```
class MyFastSimModel : public G4VFastSimulationModel
{
public:
```

MyFastSimModel.hh

```
//-----
// Constructor, destructor
//-----
```

```
MyFastSimModel (G4String, G4Region*);
MyFastSimModel (G4String);
~MyFastSimModel ();
```

**Inheritance of
G4VFastSimulationModel**

```
//-----
// Virtual methods of the base
// class to be coded by the user
//-----
```

Functions to compile:

```
// -- IsApplicable
virtual G4bool IsApplicable(const G4ParticleDefinition&);
// -- ModelTrigger
virtual G4bool ModelTrigger(const G4FastTrack &);
// -- User method DoIt
virtual void DoIt(const G4FastTrack&, G4FastStep&);
```

How to create your own Fast Simulation Model?

MyFastSimModel.cc

```
MyFastSimModel::MyFastSimModel(G4String modelName, G4Region* envelope)
: G4VFastSimulationModel(modelName, envelope)
{
}
```

Constructor

```
//.....ooo00000ooo.....ooo00000ooo.....ooo00000ooo.....ooo00000ooo
```

```
MyFastSimModel::MyFastSimModel(G4String modelName)
: G4VFastSimulationModel(modelName)
{
}
```

```
//.....ooo00000ooo.....ooo00000ooo.....()ooo00000ooo.....ooo00000ooo
```

```
MyFastSimModel::~MyFastSimModel
{
}
```

Destructor

How to create your own Fast Simulation Model: conditions

MyFastSimModel.cc

```
G4bool MyFastSimModel::IsApplicable(const G4ParticleDefinition& particleType)
{
    return
        &particleType == G4Electron::ElectronDefinition() ||
        &particleType == G4Positron::PositronDefinition() ||
        &particleType == G4Gamma::GammaDefinition();
    //& my particles ...
}
```

The model is applicable
for these particles:

```
//....ooo00000ooo.....ooo00000ooo.....ooo00000ooo.....ooo00000ooo...
```

```
G4bool MyFastSimModel::ModelTrigger(const G4FastTrack& fastTrack)
{
    //my code:
    //...
    return MyCondition;
}
```

The model is activated
at this condition

How to create your own Fast Simulation Model: your model

```
void MyFastSimModel::DoIt(const G4FastTrack& fastTrack,  
                          G4FastStep& fastStep)
```

Write your code

MyFastSimModel.cc

```
{
```

```
//get some necessary information
```

```
G4double Etotal = fastTrack.GetPrimaryTrack()->GetTotalEnergy();
```

```
G4double mass = fastTrack.GetPrimaryTrack()->GetParticleDefinition()->GetPDGMass();
```

```
G4double charge = fastTrack.GetPrimaryTrack()->GetParticleDefinition()->GetPDGCharge();
```

```
G4ThreeVector MomentumDirection=fastTrack.GetPrimaryTrackLocalDirection();
```

```
G4ThreeVector xyz = fastTrack.GetPrimaryTrackLocalPosition();
```

```
G4double TGlobal = fastTrack.GetPrimaryTrack()->GetGlobalTime();
```

```
//fastTrack.Get...
```

```
//do very important simulations
```

```
//my code ...
```

```
//set new parameters:
```

```
//set global time
```

```
fastStep.ProposePrimaryTrackFinalTime(TGlobal);
```

```
//set final position
```

```
fastStep.ProposePrimaryTrackFinalPosition(xyz);
```

```
//set final kinetic energy
```

```
fastStep.ProposePrimaryTrackFinalKineticEnergy(Etotal-  
        fastTrack.GetPrimaryTrack()->GetParticleDefinition()->GetPDGMass());
```

```
//set final momentum direction
```

```
fastStep.ProposePrimaryTrackFinalMomentumDirection(MomentumDirection);
```

```
//kill a primary particle if necessary
```

```
fastStep.KillPrimaryTrack();
```

Propose new parameters

Get parameters

You may kill the particle if necessary

```
}
```

Secondary particle production

MyFastSimModel.cc

```
void MyFastSimModel::DoIt(const G4FastTrack& fastTrack,
                          G4FastStep& fastStep)
{
    //some code ...

    //there is a default but it is better to do:
    fastStep.SetNumberOfSecondaryTracks(MaxParticlesProducedPerStep);

    //particle declaration
    const G4DynamicParticle theGamma =
        G4DynamicParticle(G4Gamma::Gamma(), PhotonMomentumDirection, Ephoton);

    //generation of a secondary photon
    fastStep.CreateSecondaryTrack(theGamma, PhotonCoordinateXYZ, PhotonGlobalTime, true);
}
```


The last step: Fast Simulation Process registration

- Add to main:

Register FastSimulationPhysics

```
G4FastSimulationPhysics* fastSimulationPhysics = new G4FastSimulationPhysics();
fastSimulationPhysics->BeVerbose();
// -- activation of fast simulation for particles having fast simulation models
// -- attached in the mass geometry:
fastSimulationPhysics->ActivateFastSimulation("e-");
fastSimulationPhysics->ActivateFastSimulation("e+");
// -- Attach the fast simulation physics constructor to the physics list:
physicsList->RegisterPhysics( fastSimulationPhysics );
```

That's it. Enjoy! :)

Important:

- If any **condition** of the model is **not fulfilled** (IsApplicable, ModelTrigger), **standard Geant4** processes will be **active** just as usual
- If there are **several Fast Simulation models**, the **first** model in the list will be **activated** for which the conditions are fulfilled

Link your model to a G4Region in DetectorConstruction

● Add to DetectorConstruction::Construct()

```
//my volume
G4Box* MySolid = new G4Box("MyBox",SizeX/2,SizeY/2,SizeZ/2.);
G4LogicalVolume* MyVolumeLogic = new G4LogicalVolume(MySolid,MyMaterial,"MyVolume");
new G4PVPlacement(Rotation,Position,MyVolumeLogic,"MyVolume",logicWorld,false,0);

//my region (necessary for the FastSim model)
fRegion = new G4Region("MyRegion");
fRegion->AddRootLogicalVolume(MyVolumeLogic);
```

Volume declaration
(completely standard)

G4Region declaration

Add Logic Volume to
G4Region declaration

● Add to DetectorConstruction::ConstructSDandField()

```
void DetectorConstruction::ConstructSDandField()
{
    // ----- fast simulation -----
    //extract the region of the crystal from the store
    G4RegionStore* regionStore = G4RegionStore::GetInstance();
    G4Region* MyRegion = regionStore->GetRegion("MyRegion");

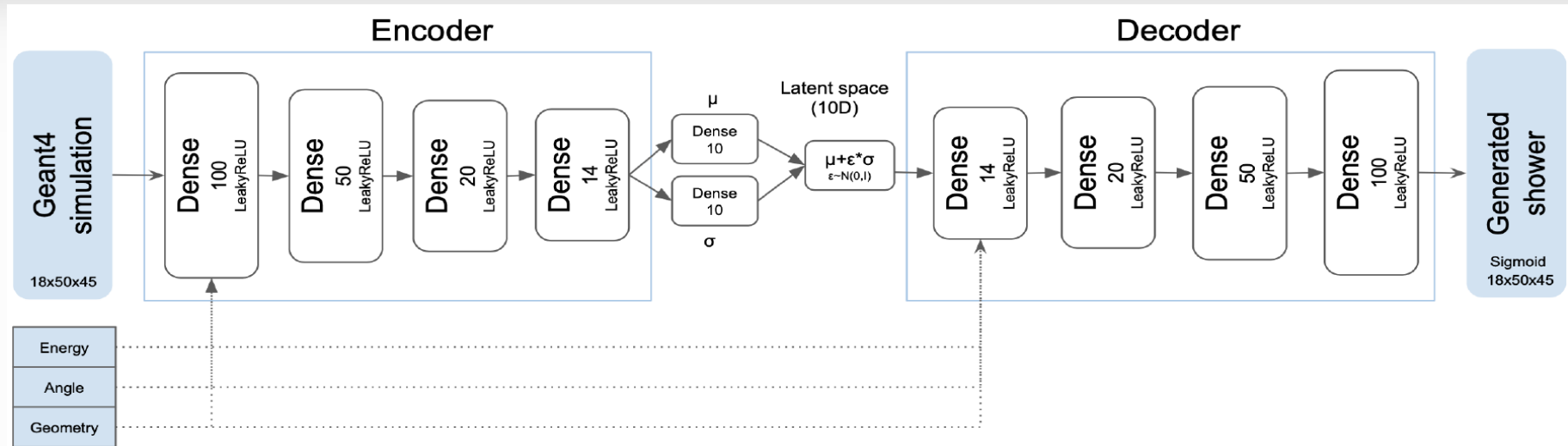
    //create the channeling model for this region
    MyFastSimModel* MyModel = new MyFastSimModel("ChannelingModel",MyRegion);

    //some options of the model...
}
```

Get G4Region

Declare your FastSim model

Applications: Machine Learning model

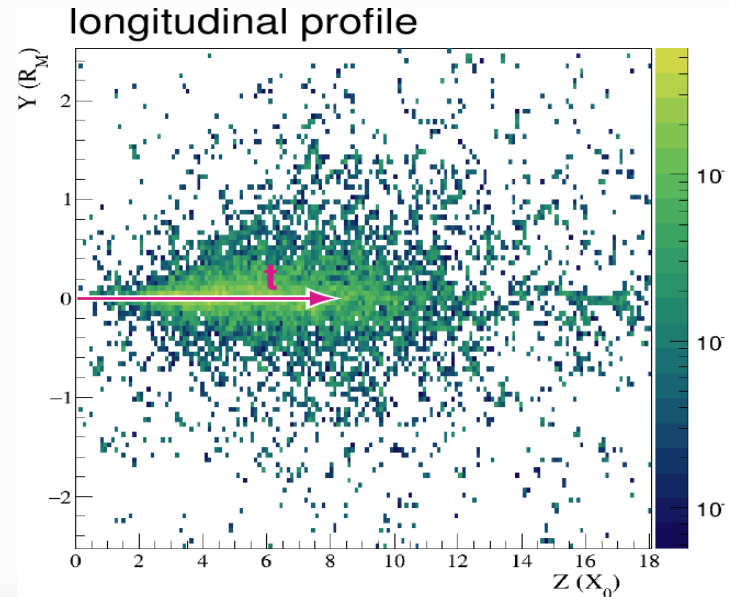
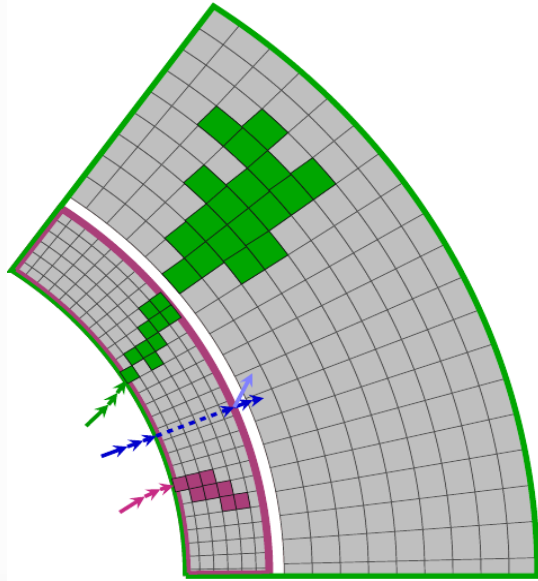


Variational autoencoder is one of the best way to randomly **generate** a distribution using initial parameters

Fast Simulation Model can upload the neural network parameters for **inference** using **Lightweight Trained Neural Network** or **Open Neural Network Exchange** libraries

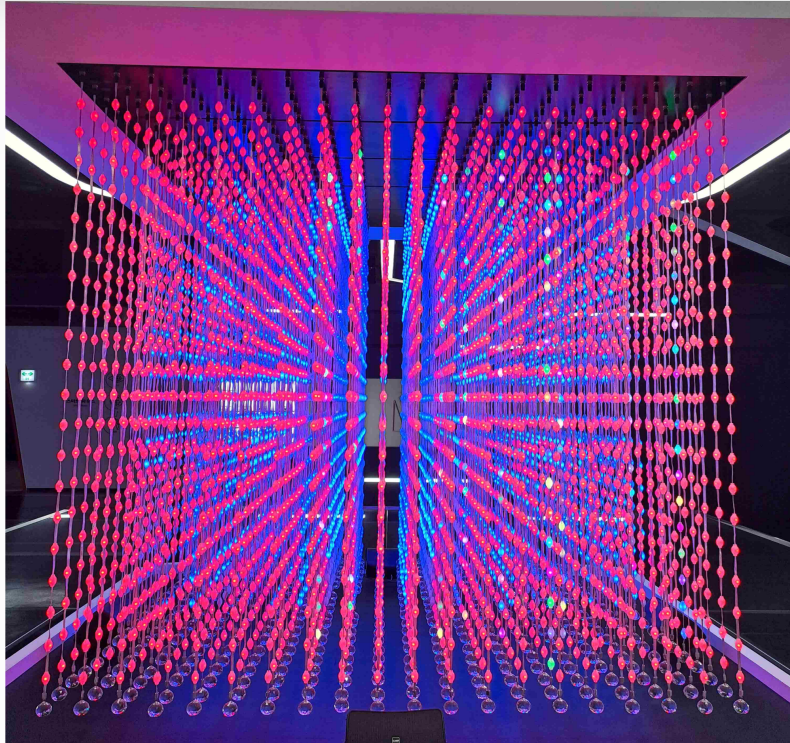
Applications: electromagnetic showers

- Simulation of **electromagnetic showers** in matter (**e.m. calorimeters**, ...)
- Simulation of **sampling calorimeters**
- **Machine Learning**
- Implementation of **external codes** into Geant4



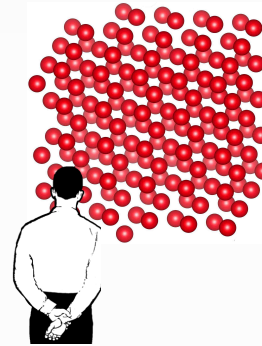
From Anna Zaborowska presentation,
[examples/extended/parameterisations/Par01](#)

Applications: G4ChannelingFastSimModel

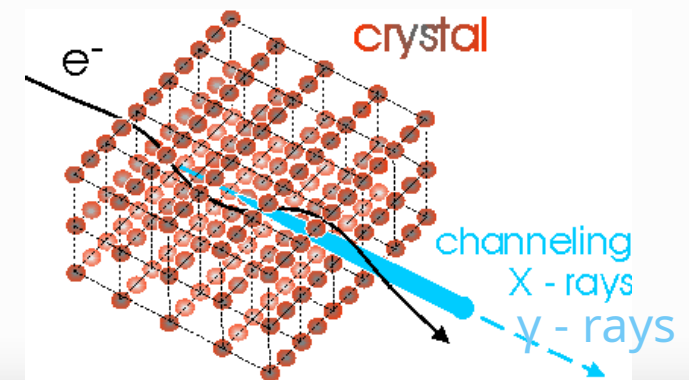
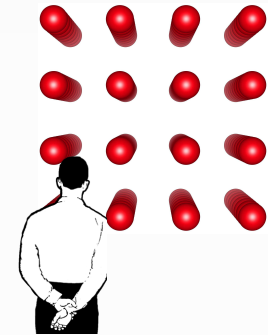


from National Science Museum,
Daejeon, Korea

Non-oriented
crystal



Oriented
crystal





Thank you for attention!