Gaseous Detector Simulation

& Physics Modelling

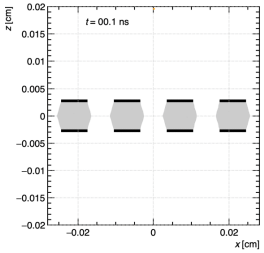
Djunes Janssens – CERN

Riccardo Farinelli – INFN sezione di Bologna

Dario Stocco – ETH Zurich

Piet Verwilligen - INFN *sezione di Bari*

inspired and grown up by many lectures of Rob Veenhof



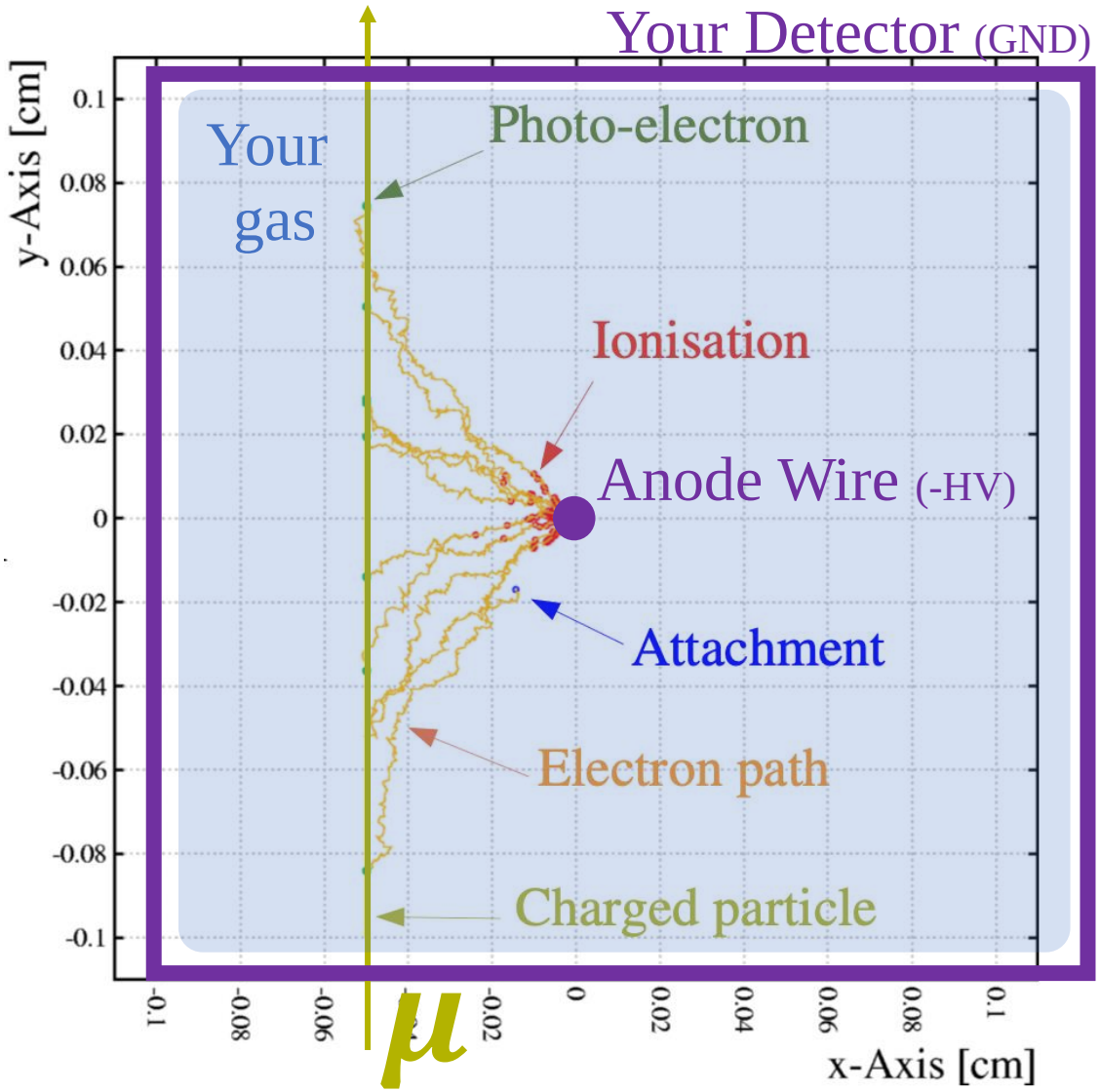
DRD1 School: Simulation Outline

- First Lecture: *the basics* *P.V. Today*
  - 0. Introduction – Garfield++
  - 1. Energy Loss – Primary Ionization
  - 2. Electric Fields in the detector
  - 3. Charged particle transport
- Second Lecture: *advanced topics* *D.J. Monday*
  - 4. Gas Gain & fluctuations
  - 5. Signal induction & Capacitive coupling
  - 6. Current limitations and Perspectives
- Hands-on Exercises:
  - SIM1: several simple exercises on the individual topics above
  - SIM2: full exercise: simulation of RPC or Resistive Micromegas

First Lecture

detector simulation basics

0. Intro: All steps: from μ to signal



Primary Ionization 1
- *Energy loss - gas*
- *Cluster distribution*

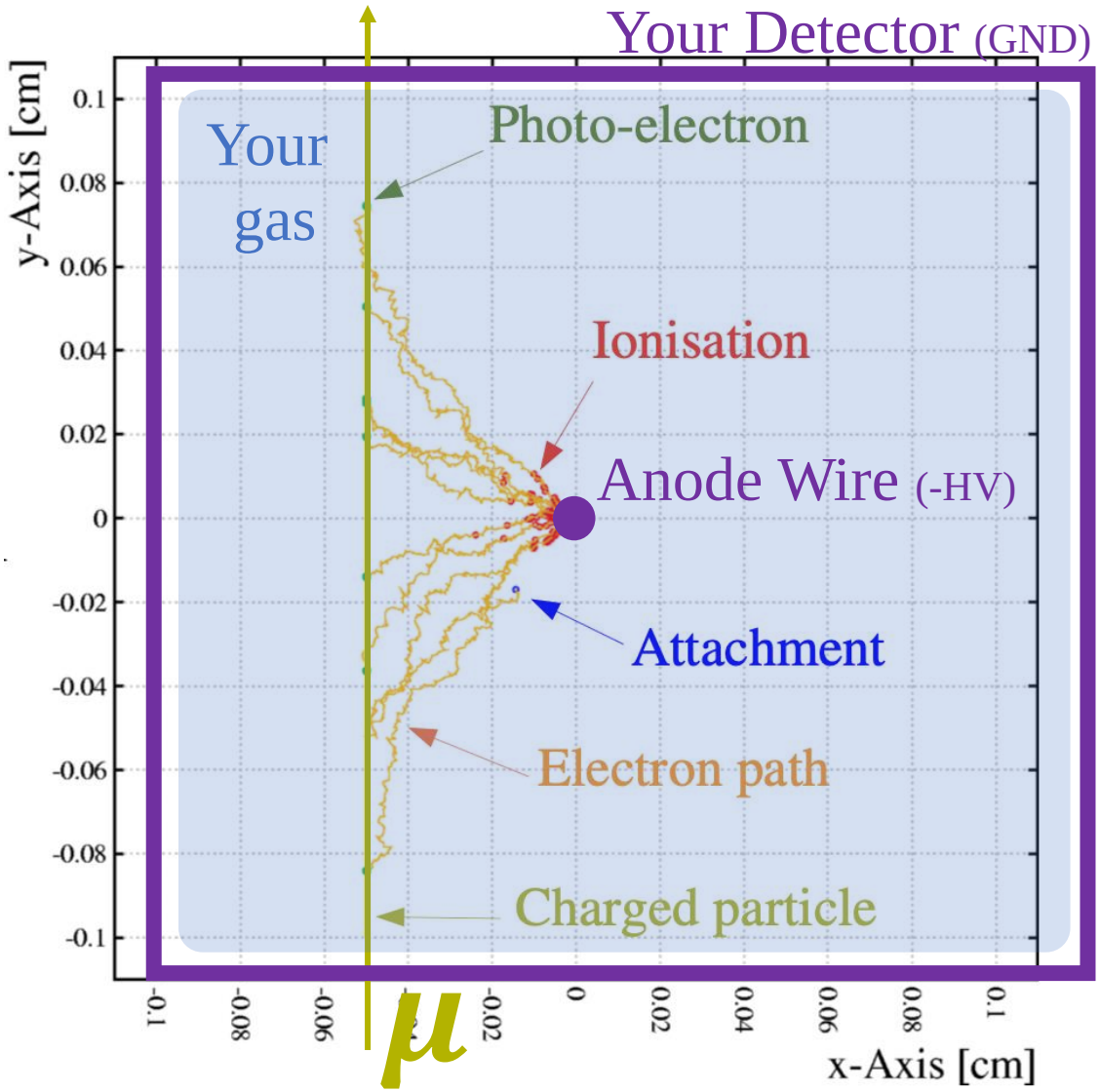
Your Detector 2
- *Geometry*
- *Electric Fields*

Charge transport 3
- *Drift & diffusion*
- *Gas cross-sections*

Charge Amplification 4
- *Townsend avalanche*
- *Avalanche fluctuations*

Signal Induction 5
- *Ramo-Shockley theorem*
- *Signal processing*

0. Intro: All steps: from μ to signal



Primary Ionization 1
- *Theory: F.Sauli, today*
- *Simulation: P.V. , today*

Your Detector 2
- *Theory: Many Lectures*
- *Simulation: P.V, today*

Charge transport 3
- *Theory: F.Sauli, today*
- *Simulation: P.V., today*

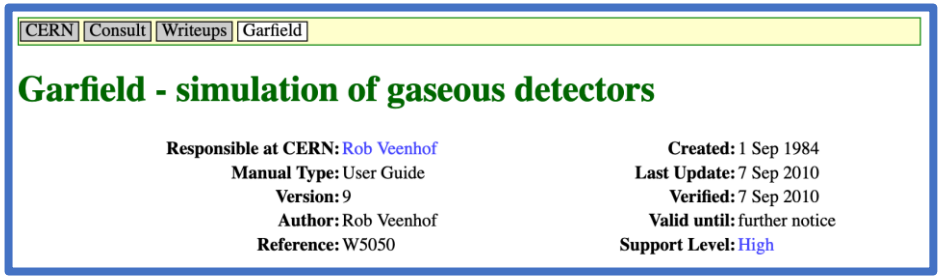
Charge Amplification 4
- *Theory: F.Sauli, today*
- *Simul: D.Janssens, Monday*

Signal Induction 5
- *Theory: W.Riegler, Friday*
- *Simulation: D.Janssens, Monday*

Plot © Rob Veenhof

0. Intro: Garfield & Garfield++

- **Garfield** = Open-source toolkit for detailed simulation of charge transport and signals in particle detectors
  - Developed by Rob Veenhof as Summer Student in 1984
  - Fortran77, continuously devel'ed & expanded 1984 - 2010
- **Garfield++** = New program written in C++ by Rob Veenhof and Heinrich Schindler (then PhD student)
  - Implemented most functionallity (not all!) from garfield
  - Can now simulate also semiconductor devices



CERN | Consult | Writeups | Garfield

Garfield - simulation of gaseous detectors

Responsible at CERN: Rob Veenhof
Manual Type: User Guide
Version: 9
Author: Rob Veenhof
Reference: W5050

Created: 1 Sep 1984
Last Update: 7 Sep 2010
Verified: 7 Sep 2010
Valid until: further notice
Support Level: High

Garfield++ | Installation | Examples | Documentation

Garfield++

About

Garfield++ is a toolkit for the detailed simulation of particle detectors. Garfield++ shares functionality with the Garfield p interface, which is based on ROOT.

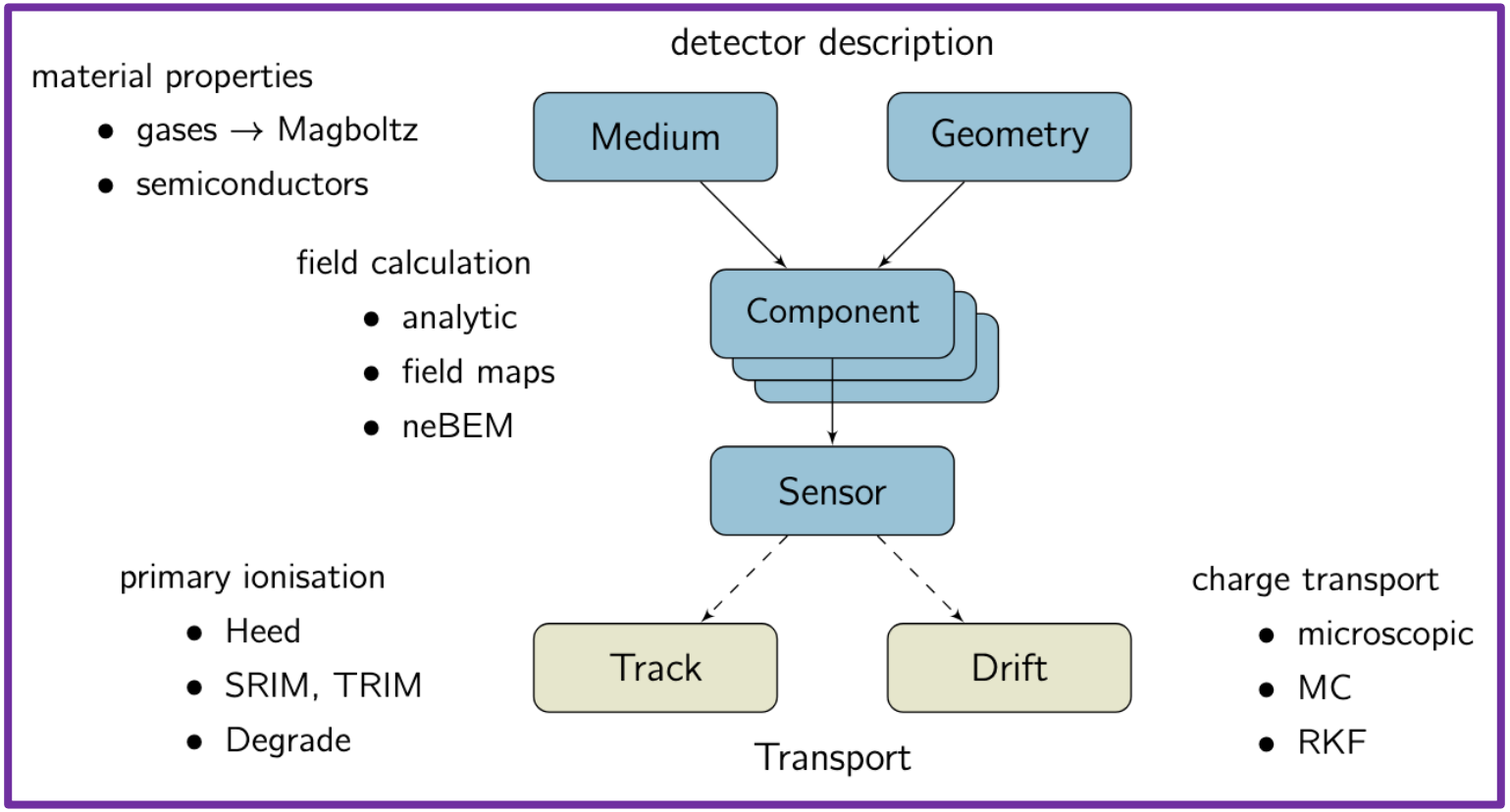
More...

Getting started

- Installation
- Examples
- Documentation (User Guide, Doxygen, FAQ)

- https://garfield.web.cern.ch/garfield/
- https://garfieldpp.web.cern.ch/garfieldpp/
- https://cds.cern.ch/record/1500583

0. Intro: Garfield++ Structure

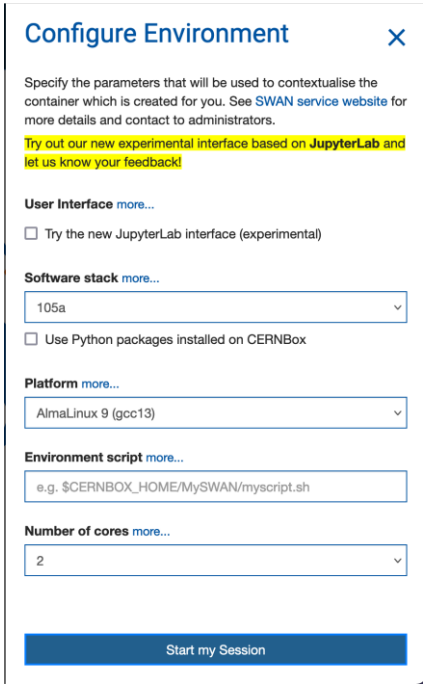


Schematic © H.Schindler

- https://garfield.web.cern.ch/garfield/
- https://garfieldpp.web.cern.ch/garfieldpp/
- https://cds.cern.ch/record/1500583

0. Intro: Garfield++ Structure

- Quickest way to start using Garfield++: **SWAN**
  - Jupyter notebook using python programming language
  - For who has a CERN computing account
  - We access directly the latest (nightly) build of Garfield++
  - go to https://swan-k8s.cern.ch
  - *An example is/will be attached to the indico page*
- Alt: on LXPlus *or* installed on a Linux Virt.Machine




```
In [1]: import ROOT
        import os, sys
        import ctypes
        import math
        import time
        import random

        Welcome to JupyROOT 6.30/04
```

The minimum needed:

- Import ROOT
- Import OS and System commands
- Import C++ type objects

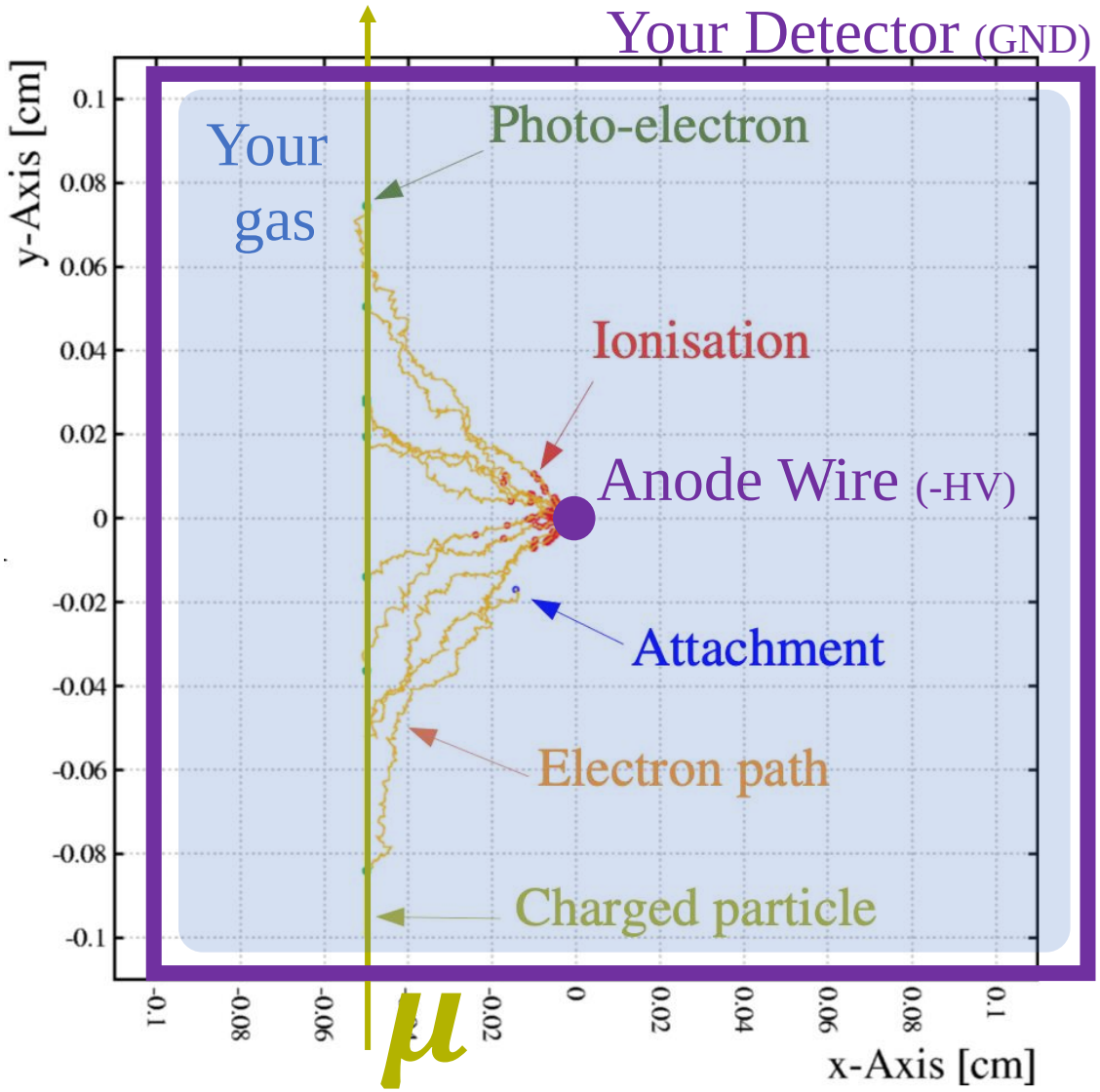
```
In [2]: # in SWAN we can access directly the Garfield++ nightly build
        path = os.path.expandvars('$ROOT_LCG_VIEW_PATH/$ROOT_LCG_VIEW_NAME/$ROOT_LCG_VIEW_PLATFORM')
        print('Path for Garfield++ software: ', path)
        ROOT.gSystem.Load(path + '/lib64/libmagboltz.so')
        ROOT.gSystem.Load(path + '/lib64/libGarfield.so')
        os.environ['HEED_DATABASE'] = path + '/share/Heed/database'

        Path for Garfield++ software:  /cvmfs/sft.cern.ch/lcg/views/LCG_105a_swan/x86_64-el9-gcc13-opt
```

Load the path to the nightly build

Load the paths of Garfield Libs

0. Intro: What is not (yet) modelled?



Primary Ionization

- *HEED assumes zero Energy loss*
- *Interactions in detector material*

Your Detector

- *Non-smooth surfaces*
- *Error on Electric Field calc*

Charge transport

- *Correct drift velocity for ions*

Charge Amplification

- *A correct limit to max charge amplification*
- *Emission of UV-photons from excited states*
- *Smooth transition from Micr.Trk to Swarm*

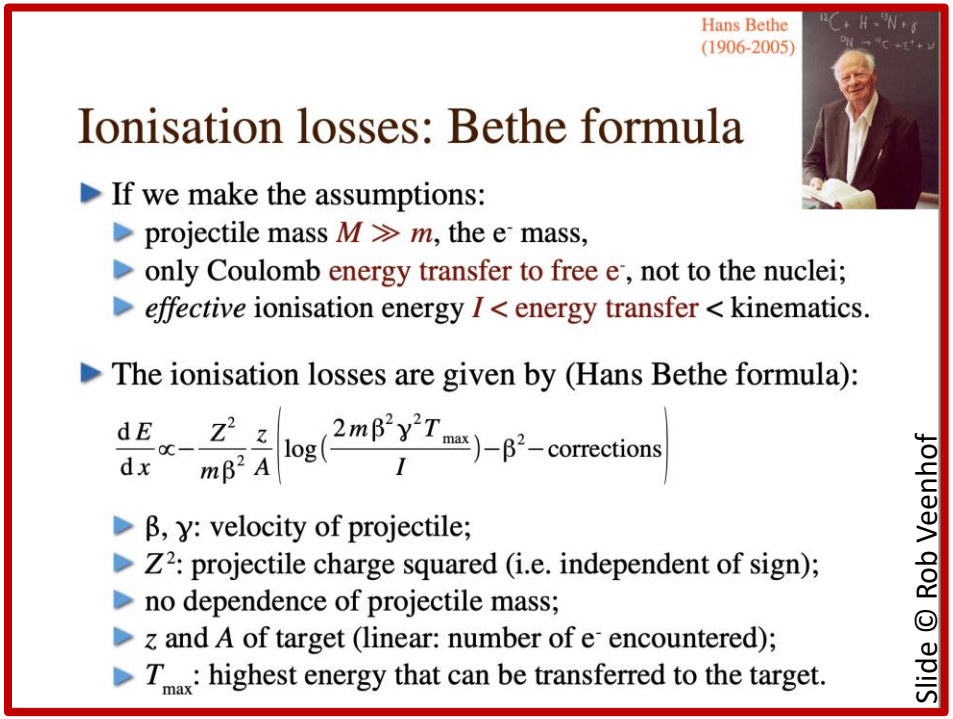
Signal Induction

- *New: effects of Resistive materials*
- *Transmission line effects*

Plot © Rob Veenhof

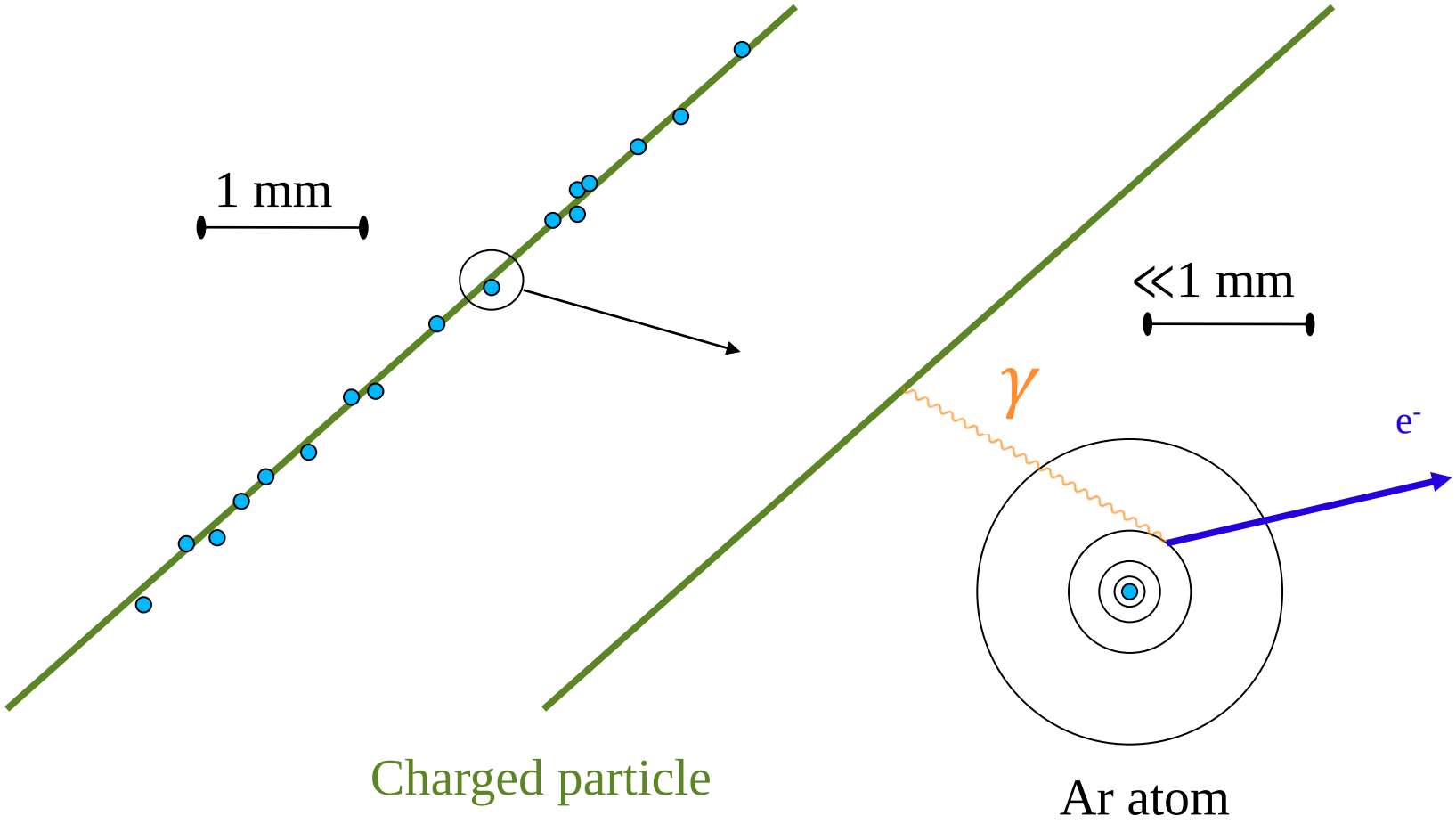
1. Interaction of particles with Matter

- Bethe-Bloch
  - For charged particles
  - *Tells us only about E-loss*
  - *Not about distribution along the primary track*



- Interaction of relativistic charged particles
  - Described in "Photo-Absorption Ionisation (& Relaxation) PAI(R)
    - GEANT4 uses PAI, HEED (interfaced to Garfield++) implements PAIR
- Interaction of slow, heavy charged particles
  - Stopping and Range of Ions in Matter (SRIM)
- Interaction of neutral particles
  - Interaction of Photons (HEED) & Neutrons (GEANT – *not covered*)

1. Interaction of particles w/ matter



Slide © Rob Veenhof

1. formulae PAI(R) model

- Key ingredient: photo-absorption cross section $\sigma_\gamma(E)$

$$\frac{\beta^2\pi}{\alpha}\boxed{\frac{d\sigma}{dE}}=\frac{\sigma_\gamma(E)}{E}\log\left(\frac{1}{\sqrt{(1-\beta^2\epsilon_1)^2+\beta^4\epsilon_2^2}}\right)+\frac{1}{\sqrt{\hbar c}}\left(\beta^2-\frac{\epsilon_1}{|\epsilon|^2}\right)\theta+\frac{\sigma_\gamma(E)}{E}\log\left(\frac{2m_e c^2\beta^2}{E}\right)+\frac{1}{E^2}\int_0^E\sigma_\gamma(E_1)\,dE_1$$

Cross section to transfer energy E

Relativistic rise

Čerenkov radiation

Resonance region

Rutherford scattering

PAIR model implemented in HEED

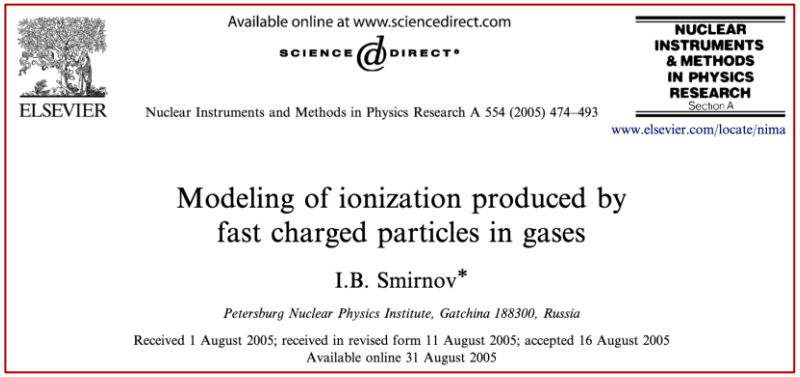
With:

$$\epsilon_2(E)=\frac{N_e\hbar c}{EZ}\sigma_\gamma(E)$$

$$\epsilon_1(E)=1+\frac{2}{\pi}\mathrm{P}\int_0^\infty\frac{x\epsilon_2(x)}{x^2-E^2}dx$$

$$\theta=\arg(1-\epsilon_1\beta^2+i\epsilon_2\beta^2)=\frac{\pi}{2}-\arctan\frac{1-\epsilon_1\beta^2}{\epsilon_2\beta^2}$$

I.B. Smirnov, Nucl.Instr.Meth.A 554 (2005) 474


Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Nuclear Instruments and Methods in Physics Research A 554 (2005) 474–493
NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH Section A
www.elsevier.com/locate/nima

Modeling of ionization produced by fast charged particles in gases

I.B. Smirnov*

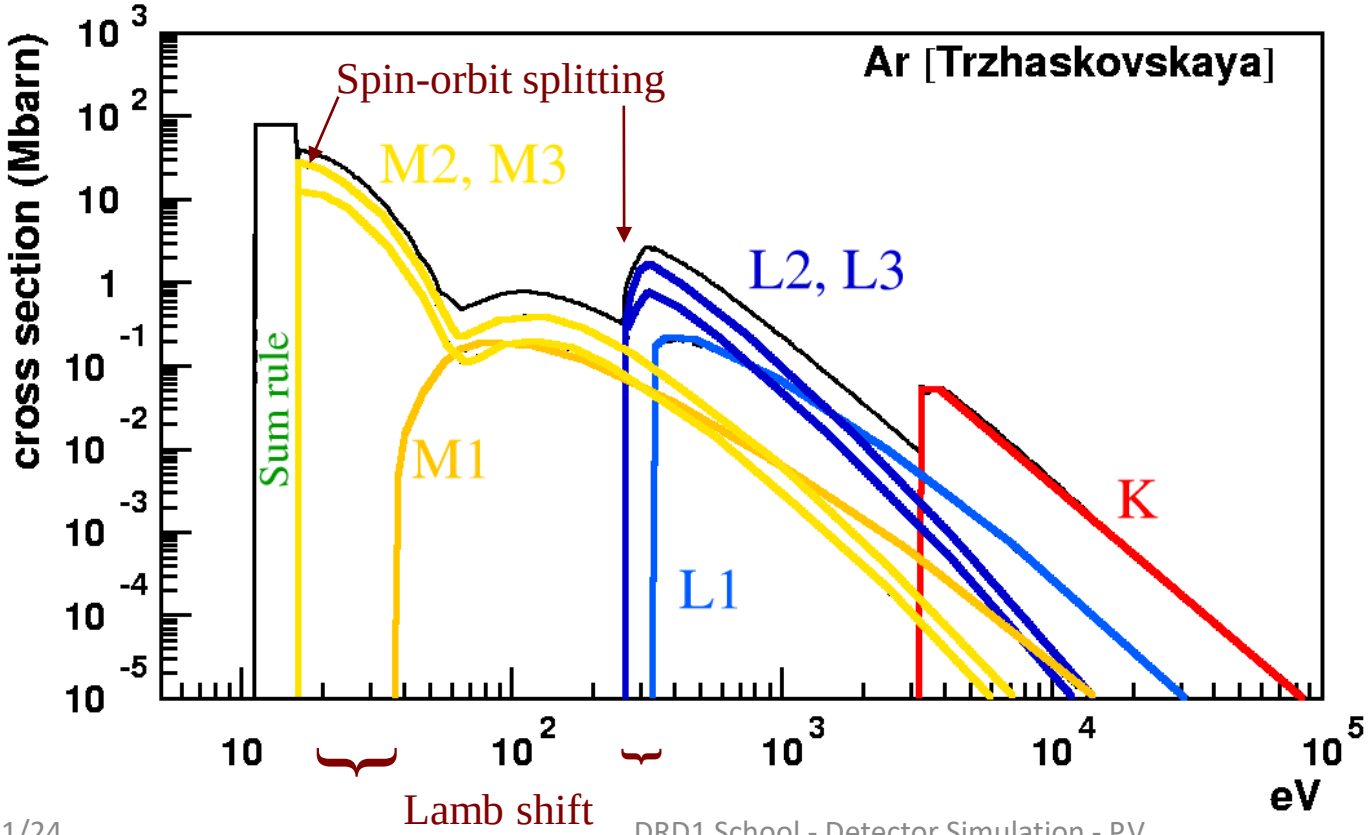
Petersburg Nuclear Physics Institute, Gatchina 188300, Russia

Received 1 August 2005; received in revised form 11 August 2005; accepted 16 August 2005
Available online 31 August 2005

Slide © Rob Veenhof

1. Photo-absorption in argon

- Argon has 3 shells, hence 3 groups of lines:



K = 1s

L1 = 2s
L2 = 2p 1/2
L3 = 2p 3/2

M1 = 3s
M2 = 3p 1/2
M3 = 3p 3/2

Plot © Igor B. Smirnov - Slide © Rob Veenhof

1. Primary Interactions / Clusters

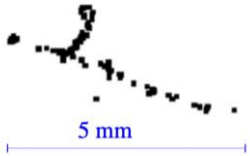
- **Ionisation electrons** deposited in Ar
  - dE/dx divided by ionisation potential (IP): 160e-/cm
  - Heed: 72 e-/cm – Degrad 62 e-/cm
- On average need more energy than binding energy to ionize an electron
  - Some energy goes into excitations (W)
  - Not all energy is used (F)

- Energy is liberated in **primary clusters**
  - Not evenly spaced – not even exponential

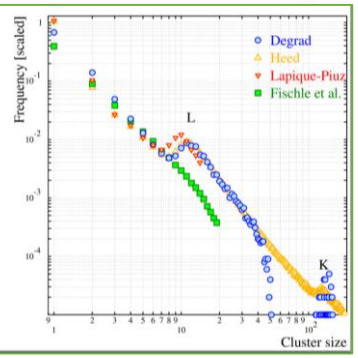
25-30 clusters/cm

- ***δ* electrons**

▶ Deposits are not always "lumps":



▶ Substantial variations between models and compared with data.



References:
F. Lapique and F. Piuz (1980) 10.1016/0029-554X(80)90744-2
H. Fischle, J. Heintze and B. Schmidt (1991) 10.1016/0168-9002(91)90460-8
Heed: I.B. Smirnov (2005) 10.1016/j.nima.2005.08.064
Degrad: S.F. Biagi (2016)

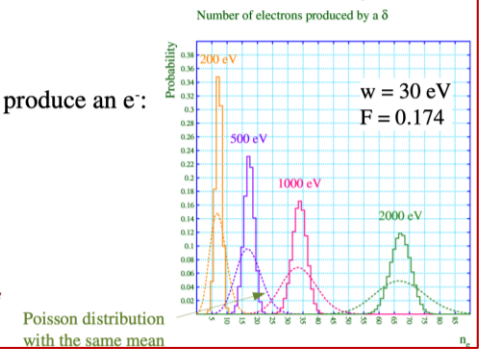
▶ Distribution of the number of secondary electrons n_e created by a photo- or Auger-electron with energy E_δ:

▶ mean of n_e:
work w needed to produce an e⁻:

$$\bar{n}_e = E_\delta / w(E_\delta)$$

▶ spread of n_e:
Fano factor F:

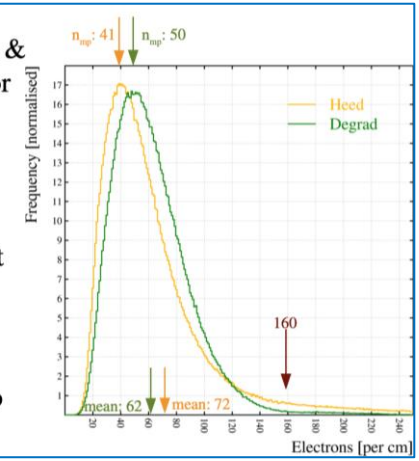
$$\sigma^2(n_e) = F(E_\delta)\bar{n}_e$$



▶ Heed, a photo-absorption & ionisation model, finds for a minimum ionising μ±:
- ▶ Peak: n_e = 41/cm
- ▶ "Mean": n_e = 72/cm

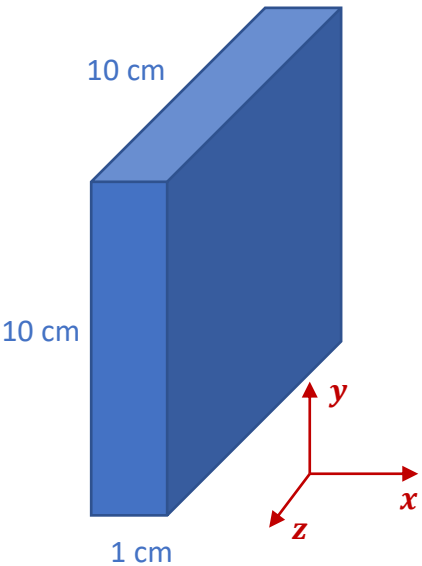
▶ Degrad, an e⁻ transport program, finds for an e⁻ at the same βγ:
- ▶ Peak: n_e = 50/cm
- ▶ "Mean": n_e = 62/cm

▶ Mean is ill-defined due to rare but large deposits.



Slides © Rob Veenhof

1. Interaction of a Charged Particle



As first step we setup a gas medium

```
gas = ROOT.Garfield.MediumMagboltz()
gas.SetTemperature(293.15) # Kelvin
gas.SetPressure(760.)      # Torr - 760 Torr = 1 bar
gas.SetComposition("ar", 70., "co2", 30.)
```

```
True

MediumMagboltz::SetComposition: Ar/CO2 (70/30)
```

```
sensor = ROOT.Garfield.Sensor()
compnt = ROOT.Garfield.ComponentConstant() # gas cannot be linked directly to the sensor - we need a component
compnt.SetArea(0, -5, -5, 1, 5, 5)         # even if 3D the method is called SetArea(x1,y1,z1,x2,y2,z2) - in cm
# compnt.SetElectricField(100,0,0)         # we set a potential of 100V over 1cm => E = 100V/cm
```

```
compnt.SetMedium(gas)                      # now link gas to this component
sensor.AddComponent(compnt)                # and link component to the sensor
```

Now we can define our particle that will interact in the gas medium

```
track = ROOT.Garfield.TrackHeed(sensor)    # we link the HEED class to calculate a track to our sensor
track.SetParticle("mu-")                   # a (negative) muon
track.SetMomentum(120e9)                   # with 120 GeV/c momentum
track.Initialise(gas,True)                 # initialize
```

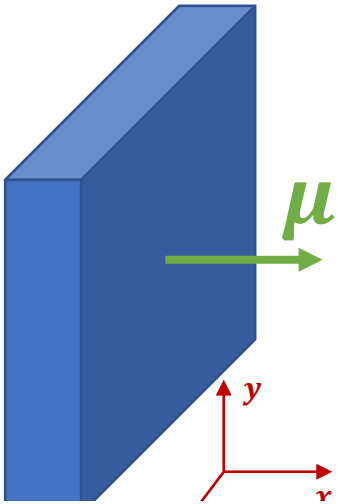
```
True

TrackHeed::Initialise:
                                                    d/install/share/Heed/database/
TrackHeed::Initialise:
    Cluster density:              38.2767 cm-1
    Stopping power (restricted): 3.07894 keV/cm
    Stopping power (incl. tail): 4.57618 keV/cm
    W value:                      28.38 eV
    Fano factor:                  0.215
    Min. ionization potential:    13.79 eV
```

Heed calculated, based on particle and the gas medium:

- N_{cls}/cm = 38 cm^{-1}
- dE/dx = 4.58 keV/cm
- W = 28.38 eV *per e-ion pair*

1. Interaction of a Charged Particle

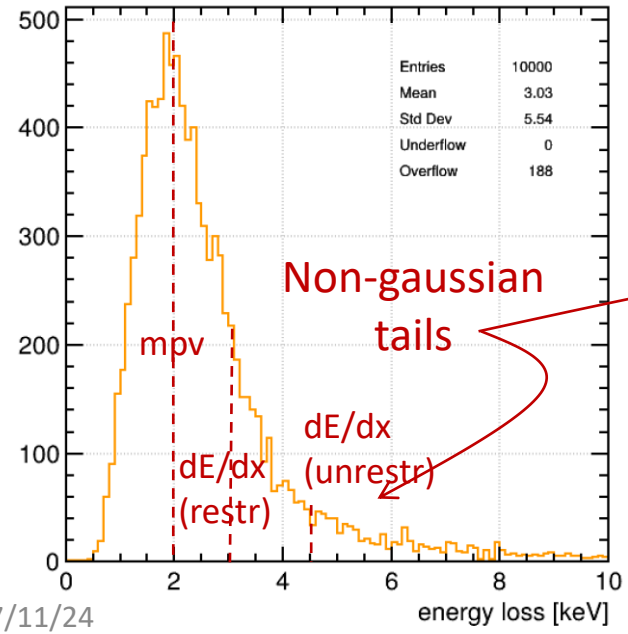


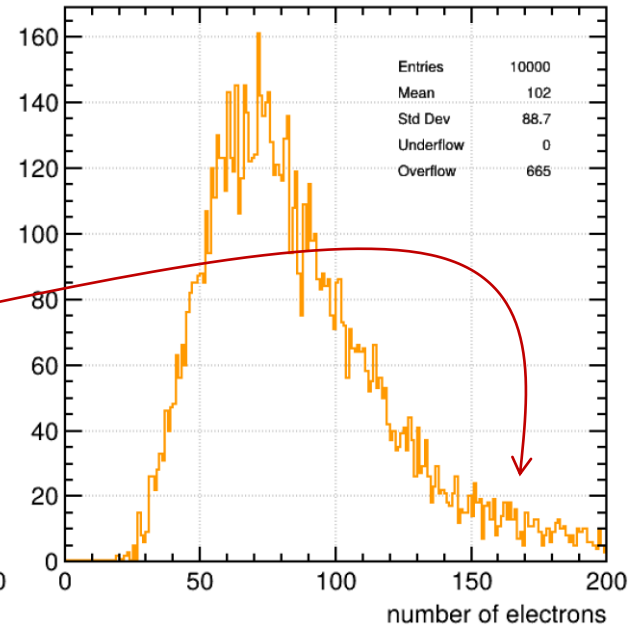
Let's send a particle through our medium

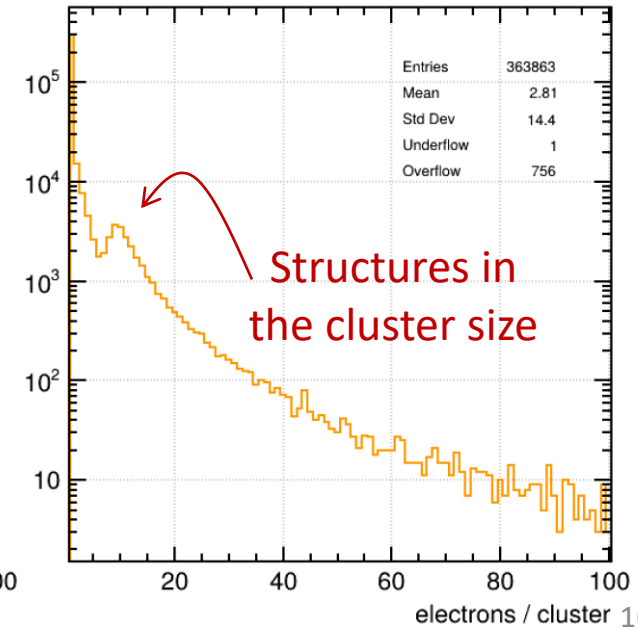
```
track.NewTrack(0,0,0,0,1,0,0)                # NewTrack(x0,y0,z0,t0,dx0,dy0,dz0)
```

True

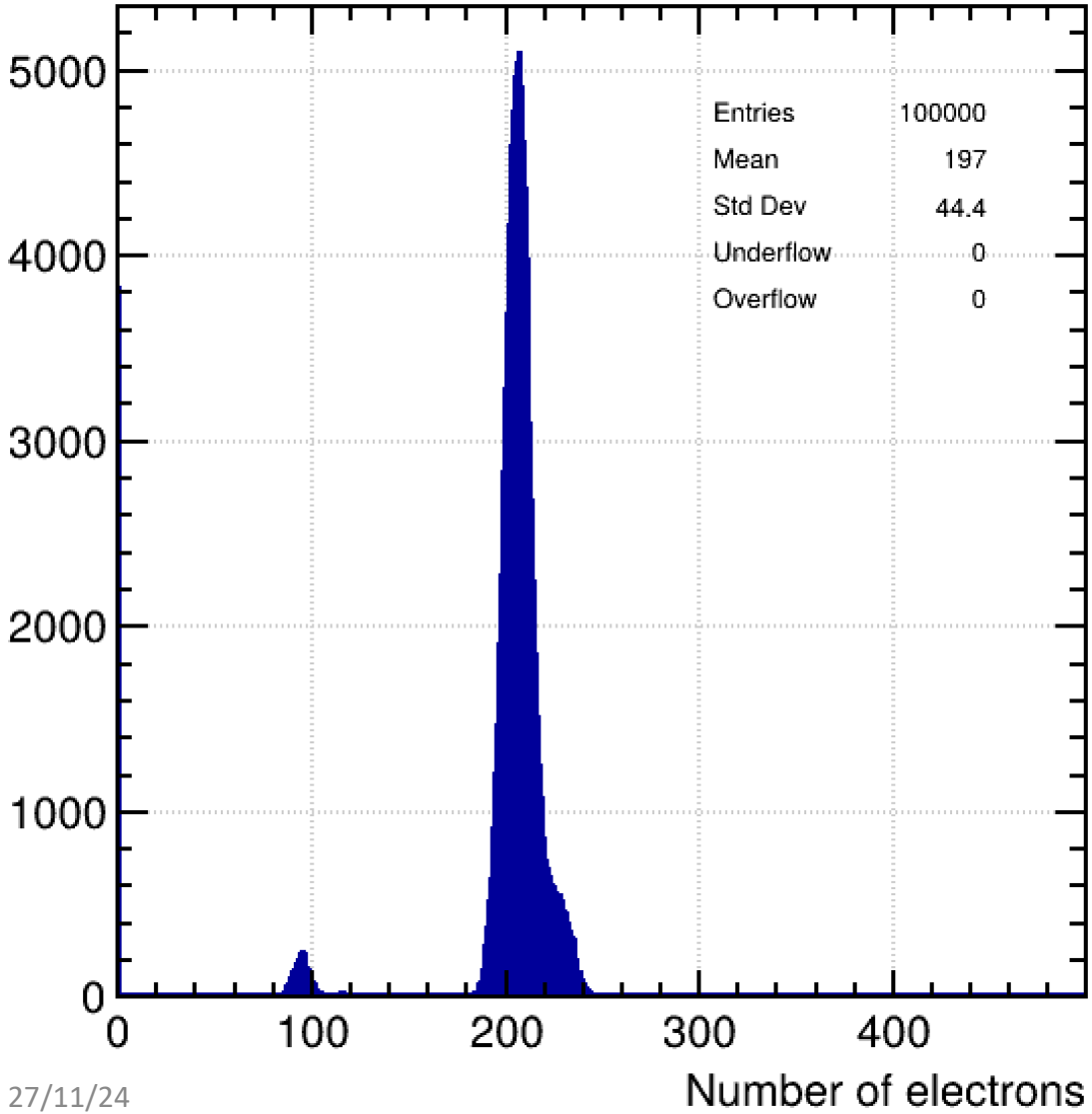
```
myclscnt = 0
edeposit = 0
totaltrk = 0
for cluster in track.GetClusters():      # track.GetClusters() returns a std::vector
    myclscnt += 1                        # the cluster object is a C++ struct
    edepclst = cluster.energy            # this is object of the struct, not a method
    nelescls = cluster.electrons.size()
    edeposit += edepclst
    totaltrk += nelescls
    if(myclscnt<5 or nelescls>1):
        print(" - Cluster ", myclscnt, "Edep = ", edepclst, "eV, Nele = ", nelescls)
print("Track: total Edep = ", edeposit, "eV, Ele = ", totaltrk, "Cls = ", myclscnt)
```







1. Interaction of a Photon / X-ray



ss - by - reference

```
types for pass-by-reference
```

```
nk the HEED class to calculate a track to our sensor
portPhoton(x0,y0,z0,t0,e0,dx0,dy0,dz0,ne,ni,np)
ni, "Number of fluorescence photons = ", np)
```

```
) Number of fluorescence photons =  c_int(0)
```

A γ with 5.9 keV interacted in the gas and produced:

- *208 electrons & 204 ions (?)*

```
nk the HEED class to calculate a track to our sensor
portPhoton(x0,y0,z0,t0,e0,dx0,dy0,dz0,ne,ni,np)
ni, "Number of fluorescence photons = ", np)
```

```
mber of fluorescence photons =  c_int(1)
```

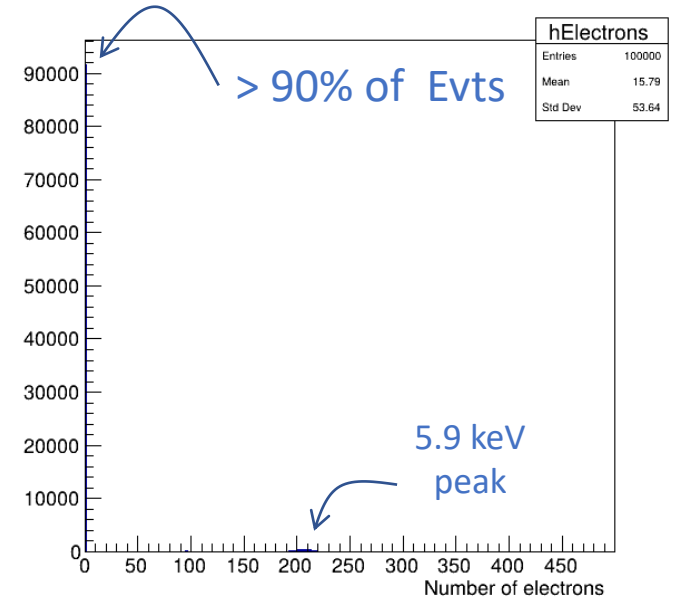
Not all γ (5.9 keV) interact in 1cm of gas

1. Interaction of a Photon / X-ray

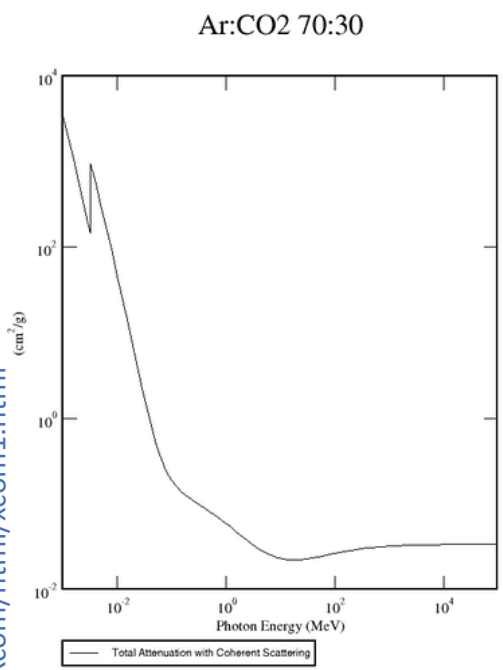
Be critic towards your results – *perform X-checks*

A Student simulated a 8 keV γ in a LEM

- (2.5mm drift gap)
- and found > 90% of Events with 0 electrons



XCOM NIST National Institute of Standards and Technology Physical Meas. Laboratory



https://physics.nist.gov/PhysRefData/Xcom/html/xcom1.html

Perform a back-of-the-envelope calculation

- *NIST XCOM database: photon cross sections for single elements, compounds and mixtures*
- *σ (@ 8keV) dominated by Photo-Electric Effect*
- *σ = 8.47 cm^2/g*
- Multiply with ρ = 1.83E-3 g/cm^3 for Ar:CO_2 70:30 gas
- $I = I_0 \exp(-\mu x)$ (Lambert-Beer) with $\mu = \sigma\rho$
- For x = 2.5 mm => only 4% interacts

2. Electric Fields & Det Geometry

- Closed expressions, "analytic method":
  - almost all 2d structures of wires, planes + periodicities;
  - dielectrics and space/surface charge are laborious;
  - fast and precise, if applicable.

  OK for parallel plate & wire detectors

  Implemented in **Garfield**

- Finite element method:
  - 2d and 3d structures, with or without dielectrics;
  - several major intrinsic shortcomings.

  Needed for MPGDs

  ANSYS, COMSOL, ELMER

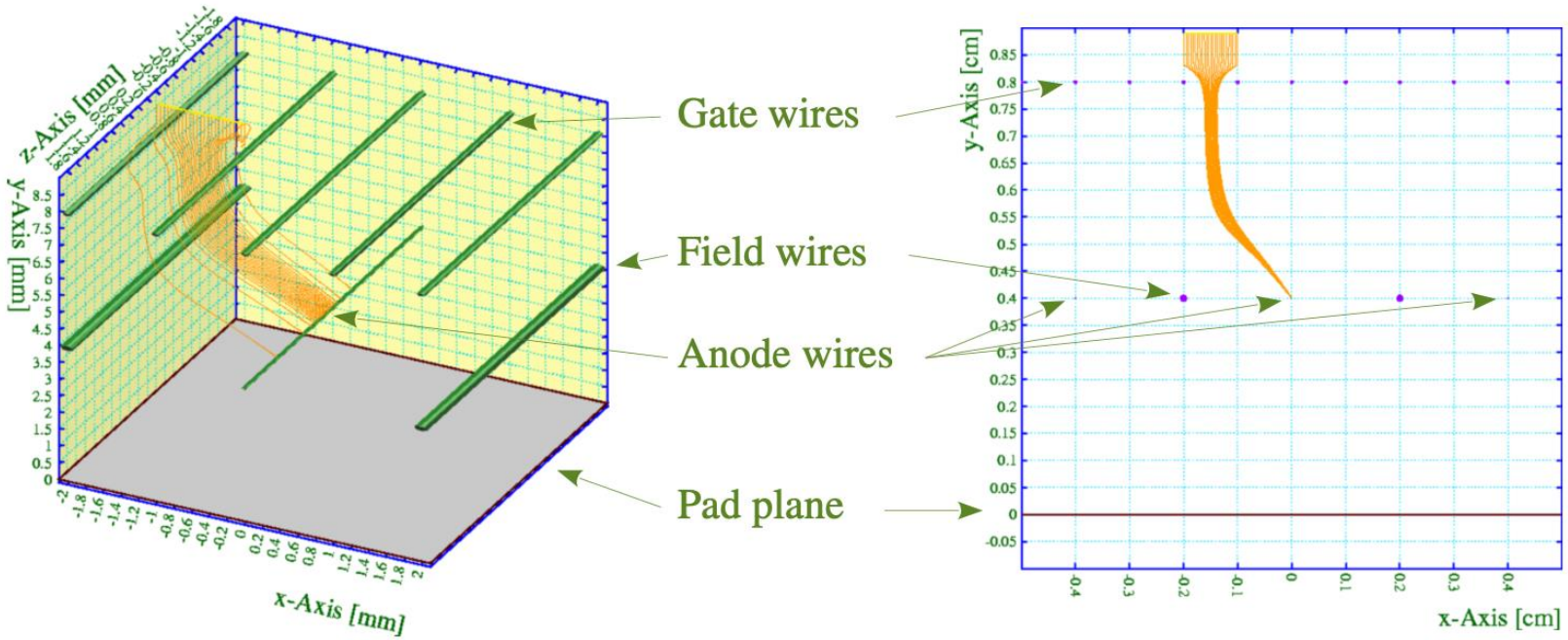
- Integral equations or Boundary element methods:
  - equally comprehensive without the intrinsic flaws;
  - technically challenging and emerging;
  - consumes more CPU time than FEM, but catching up.

  NeBEM

- Finite differences:
  - used for iterative, time-dependent calculations.

2. Analytic Fields in 2D - example

Layout: TPC read-out cell – only parallel wires and planes



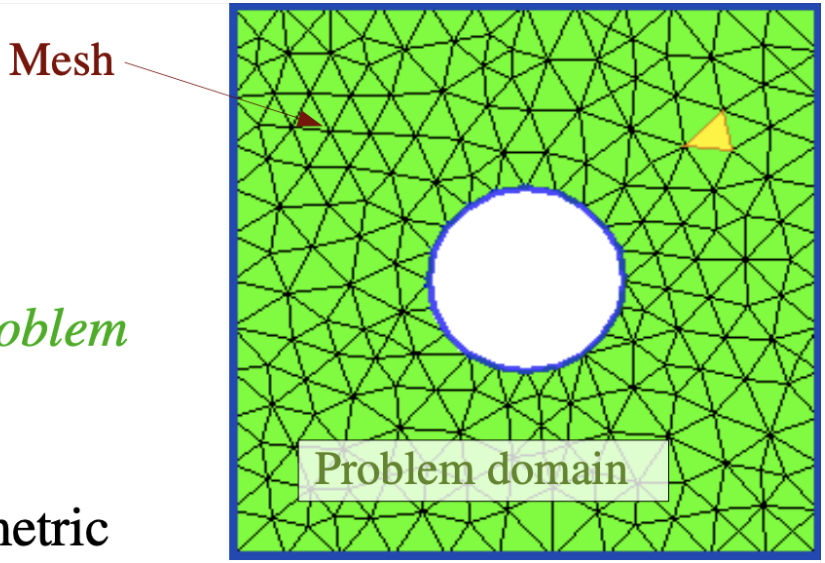
The structure repeats itself hundreds of times in x, for practical purposes we assume infinite repetition.

Garfield++ Examples:

- https://garfieldpp.web.cern.ch/garfieldpp/examples/analytic/
- https://garfieldpp.web.cern.ch/garfieldpp/examples/alicetpc/

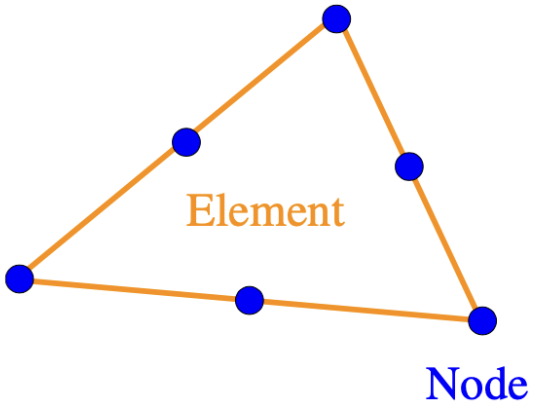
Slide © Rob Veenhof

2. Finite Element Method



Terminology

- A *mesh* subdivides the *problem domain* into *elements*.
- *Elements* are simple geometric shapes: triangles, squares, tetrahedra, hexahedra etc.
- Important points of *elements* are called *nodes*. It is usual that *nodes* are shared by several *elements*.
- The solution consists of the voltages of the *nodes*.

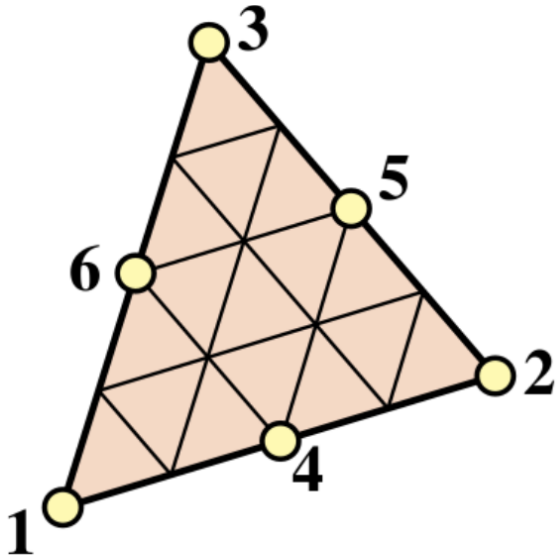


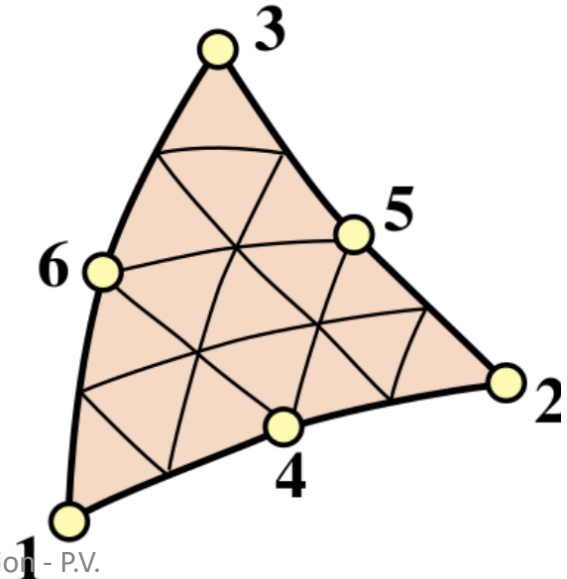
2. Finite Element Method

Element coordinates

- Elements such as triangles, tetrahedra and hexahedra are not naturally described using Cartesian coordinates.
- Instead, one uses coordinates which adapt to the element deformations.

[http://www.colorado.edu/aerospacestructures/about-cas]





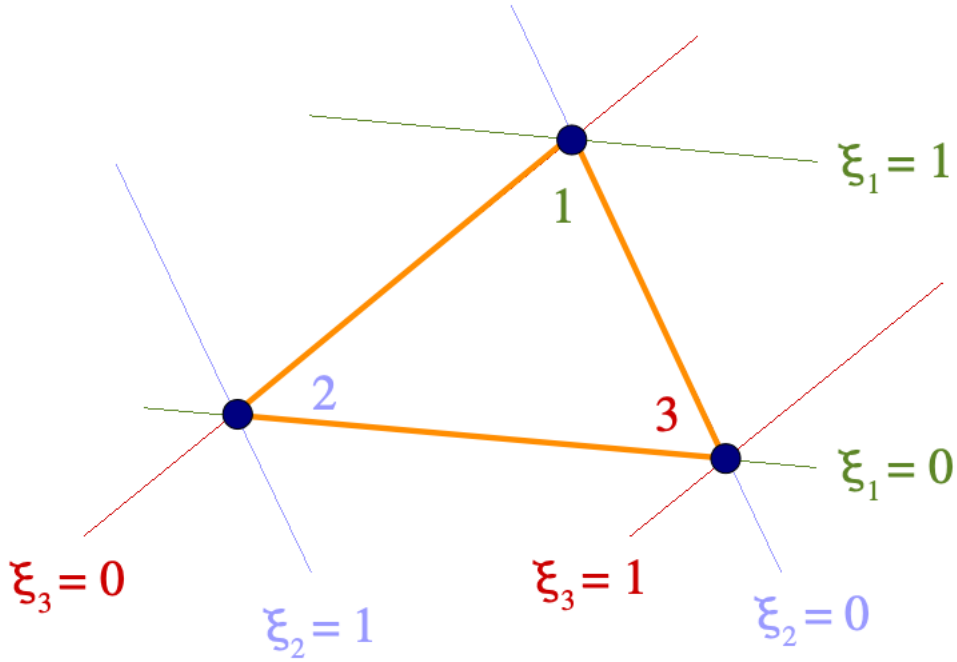
Slide © Rob Veenhof

2. Finite Element Method

Natural coordinates – example

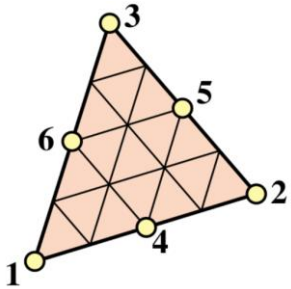
- The simplest of all elements ... the triangle.
- Its natural coordinates are (ξ_1, ξ_2, ξ_3):

$$\xi_1 + \xi_2 + \xi_3 = 1$$



2. Finite Element Method

Shape functions - interpolation

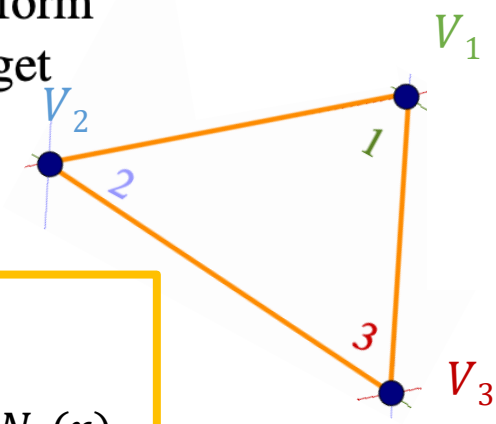


- Each node has its own *shape function* $N_i(r)$:
  - continuous functions (usually polynomial),
  - defined only in the body of the element,

  - $N_i(r) = 1$ when $r = r_i$ i.e. on node i,
  - $N_i(r) = 0$ when $r = r_j,\ i \neq j$ i.e. on all other nodes.

- The solution of a finite element problem is given in the form of the potentials v_i at each of the nodes. With these, we get the potential at interior points by interpolation:

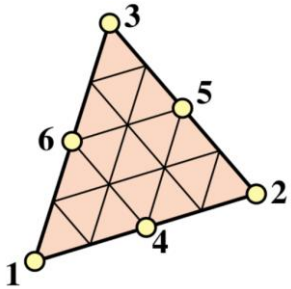
  - $V(r) = \sum v_i N_i(r)$



Example:

1st Order Triangular Element:
Nodes 1,2,3 with V_1, V_2, V_3
$V(r) = V_1 N_1(r) + V_2 N_2(r) + V_3 N_3(r)$

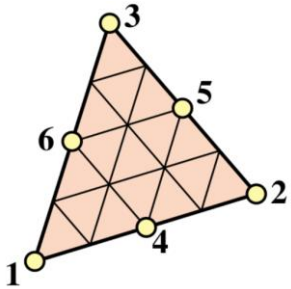
2. Finite Element Method



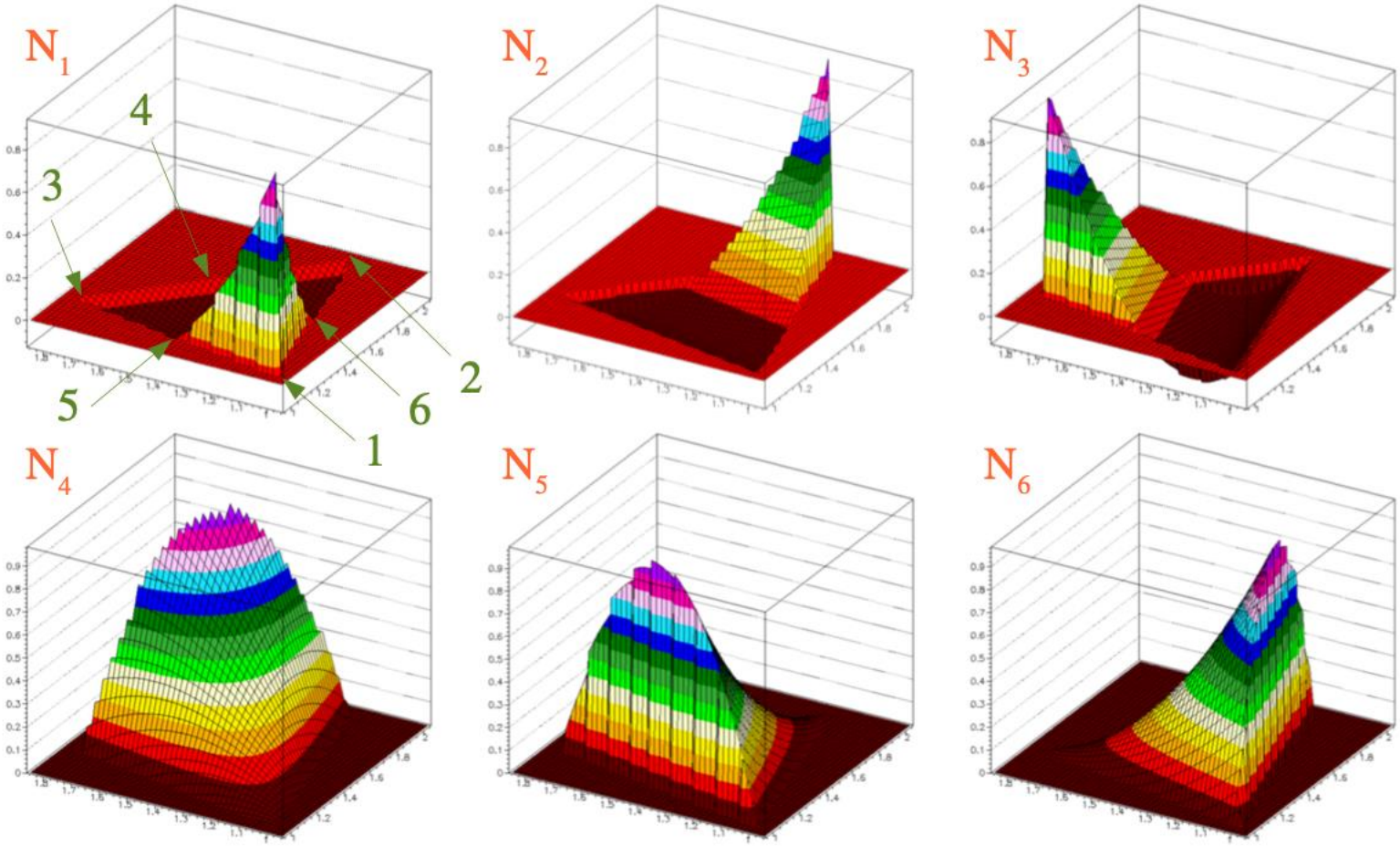
Shape functions: 2nd order triangle

- The 2nd order triangle and tetrahedron are widely used. The triangle shape functions are:
  - $N_1 = \xi_1(2\,\xi_1 - 1)$ $\quad N_4 = 4\,\xi_2\xi_3$
  - $N_2 = \xi_2(2\,\xi_2 - 1)$ $\quad N_5 = 4\,\xi_1\xi_3$
  - $N_3 = \xi_3(2\,\xi_3 - 1)$ $\quad N_6 = 4\,\xi_1\xi_2$
- The shape functions for tetrahedra are analogous. These elements too are isoparametric.
- Depending on the location of the mid-point nodes, the edges can be parabolically curved. This feature is used by e.g. Ansys but not by Maxwell.

2. Finite Element Method

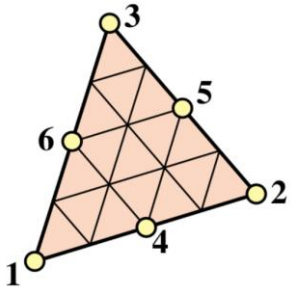


2nd Order triangle shape functions



Slide © Rob Veenhof

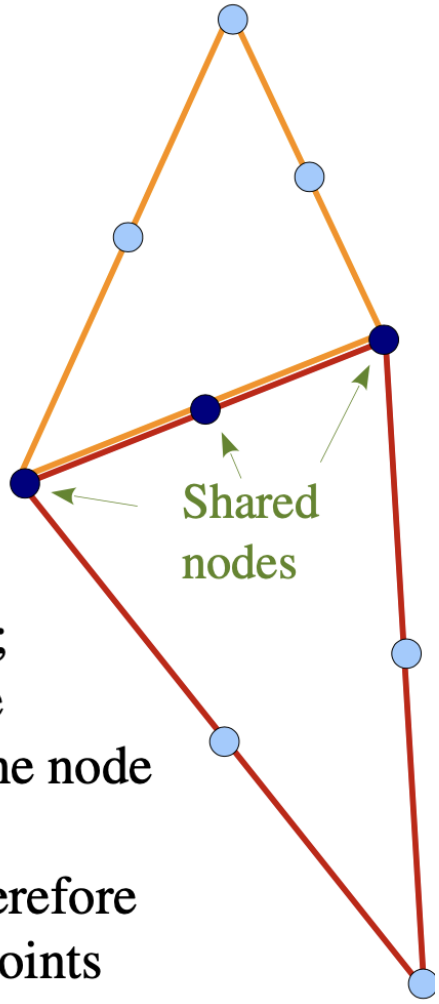
2. Finite Element Method



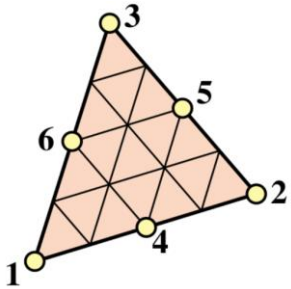
Continuity across boundaries

- Across element boundaries, the potential is guaranteed to be continuous.

- Example for a 2nd order triangle:
  - each edge shared by 2 elements, has 3 nodes;
  - the finite element method computes a unique potential for each node, i.e. the potential at the node is the same seen from both elements;
  - the potential is parabolic in each element, therefore also along each line in each element, and 3 points fully constrain a parabola.

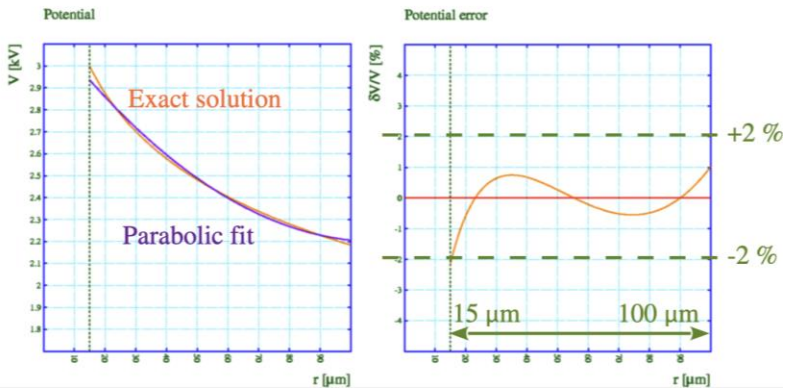


2. Finite Element Method



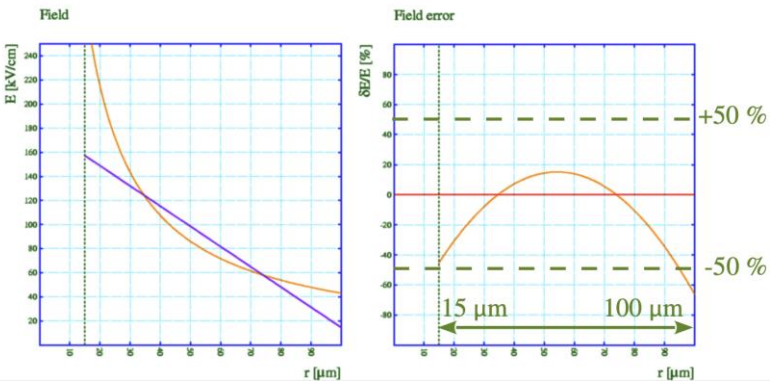
Are polynomial N_i suitable for V ?

- Polynomial shape functions imply a polynomial potential, here a 3.2 cm tube with a 30 μm wire at 3 kV inside:



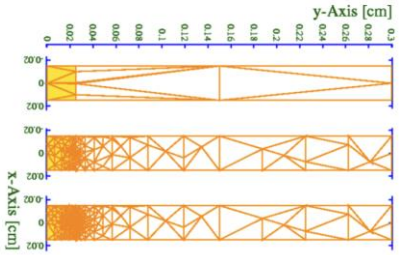
Are polynomial N_i suitable for E ?

- ... and a polynomial E field that is one order lower !



Mesh refinements

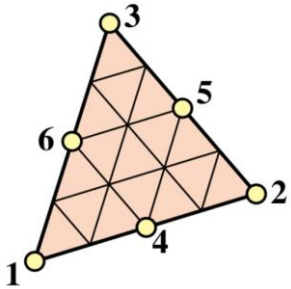
- An MSGC:
  anode: 10 × 2 μm²
  cathode: 100 × 2 μm²
  drift region: 3 mm



Food for thought ...

- The Finite Element Method is a very useful tool which can make a good engineer better, but it can make a bad engineer dangerous. [Robert D. Cook, Professor of Mechanical Engineering University of Wisconsin, Madison]
- One should wonder what the Finite Element Method can do in the hands of a physicist ...

2. Finite Element Method



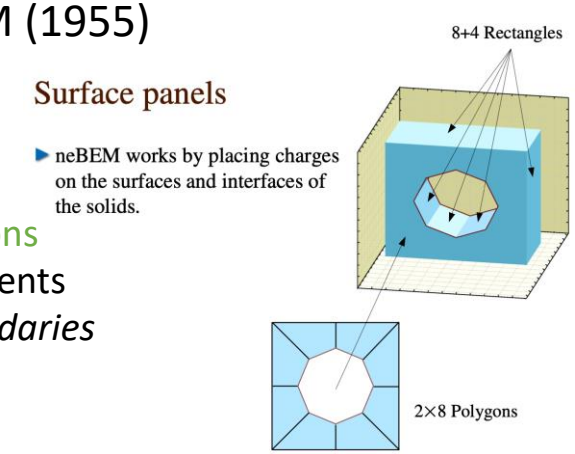
The price to pay for finite elements

- Finite element programs are flexible but they focus on the wrong thing: they solve V well, but we do not really need it:
  - quadratic shape functions do a fair job at approximating $V \approx \log(r)$ potentials;
  - potentials are continuous;
  - potentials and fields are not Maxwell compliant.

- E is what we use to transport charges, but:
  - gradients of quadratic shape functions are linear and not suitable to approximate $E \approx 1/r$, left alone $E \approx 1/r^2$ fields;
  - electric fields are discontinuous at element boundaries;
  - a local accuracy of ~50 % in high-field areas is not unusual.

2. Boundary Element Method

- Alternative approach:
  - No discretization of the domain (as in FEM)
  - But discretization of the boundaries (BEM)
- BEMS first applied around 1977 … about 20 years later than FEM (1955)
- In ad 3D detector
  - Elements are 2D surface panels located on the boundaries
  - Potentials on Electrodes => Charges on the boundary elements
  - Charge on a boundary => **Electric Field** calculated using Green's functions
  - Field in domain is superposition of fields induced by all boundary elements
  - *No discontinuities in the problem domain – but discontinuities on boundaries*
- However … method is numerically challenging, time-consuming
- neBEM: nearly-exact Boundary Element Method
  - Nearly-exact refers to how singularities in vicinity of boundaries are addressed
  - Open source, developed by our DRD1 colleagues in Saha Institute (Kolkata, IN)
- neBEM integrated in Garfield++
  - Developments still ongoing to speed up the code



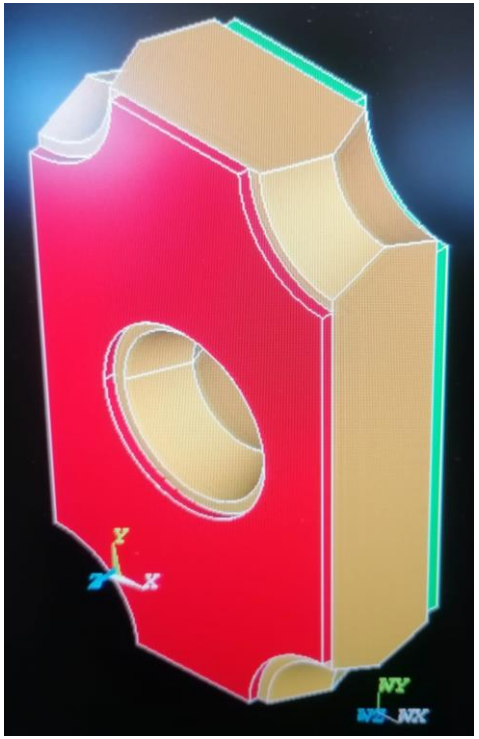
Slide © Rob Veenhof

Garfield++ Examples:

- https://garfieldpp.web.cern.ch/garfieldpp/examples/nebem/

S.Mukhodpadhyay & N.Majumdar, Computation of 3D MEMS electrostatics using a nearly exact BEM solver EABE 30 (8), 687-696, 2006

2. Electric Fields & Det Geometry



```python
compnt = ROOT.Garfield.ComponentAnsys123() # component to read in the ANSYS 3D solution (with ANSYS123 mesh el.)
# Read in the ANSYS solution files:
# - list of elements   = ELIST.lis    | info: for each element of which nodes it consist + which material
# - list of nodes      = NLIST.lis    | info: x,y,z coordinates for each node
# - list of materials  = MPLIST.lis   | info: for each material its electrical properties
# - the field solution = PRNSOL.lis   | info: voltage calculated for each node
# - last option: length units used in ANSYS to define the geometry ... can be mm, cm, um, m, ...
compnt.Initialise("Test/ELIST.lis", "Test/NLIST.lis", "Test/MPLIST.lis", "Test/PRNSOL.lis", "mm")
compnt.EnableMirrorPeriodicityX()                                          # mirror unitcell in X
compnt.EnableMirrorPeriodicityY()                                          # mirror unitcell in Y
```

```
ComponentAnsys123::Initialise:
    Read properties of 3 materials from file Test/MPLIST.lis.
    Read 8758 elements from file Test/ELIST.lis.
    Highest node number: 14066
    Background elements skipped: 0
    Read 14066 nodes from file Test/NLIST.lis.
    Read 14066 potentials from file Test/PRNSOL.lis.
ComponentAnsys123::Prepare:
    Caching the bounding boxes of all elements... done.
    Initialized tetrahedral tree.
```

- 3 materials: Gas, Cu, Kapton
- 8758 elements
- 14066 nodes
- 14066 potentials, same as nodes

```python
compnt.PrintRange()
compnt.PrintMaterials()
```

```
ComponentAnsys123::PrintRange:
        Dimensions of the elementary block
                        0 < x < 0.007          cm,
                        0 < y < 0.0121244      cm,
                     -0.1 < z < 0.1            cm,
                     -200 < V < 350            V.
        Periodicities
            x: mirror with length 0.007 cm
            y: mirror with length 0.0121244 cm
```

Dimensions of the Elementary Cell

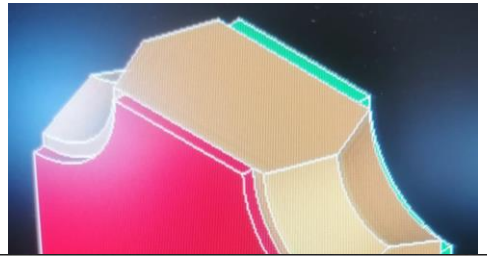
```
ComponentAnsys123::PrintMaterials:
    Currently 3 materials are defined.
      Index Permittivity  Resistivity Notes
          0        1e+10           0
          1            1          -1 (drift medium)
          2            4          -1
```

Index of the Materials:

- 0 = Cupper ($\varepsilon_r = \infty$, $\rho = 0$)
- 1 = Gas ($\varepsilon_r = 1$)
- 2 = Kapton ($\varepsilon_r = 4$)

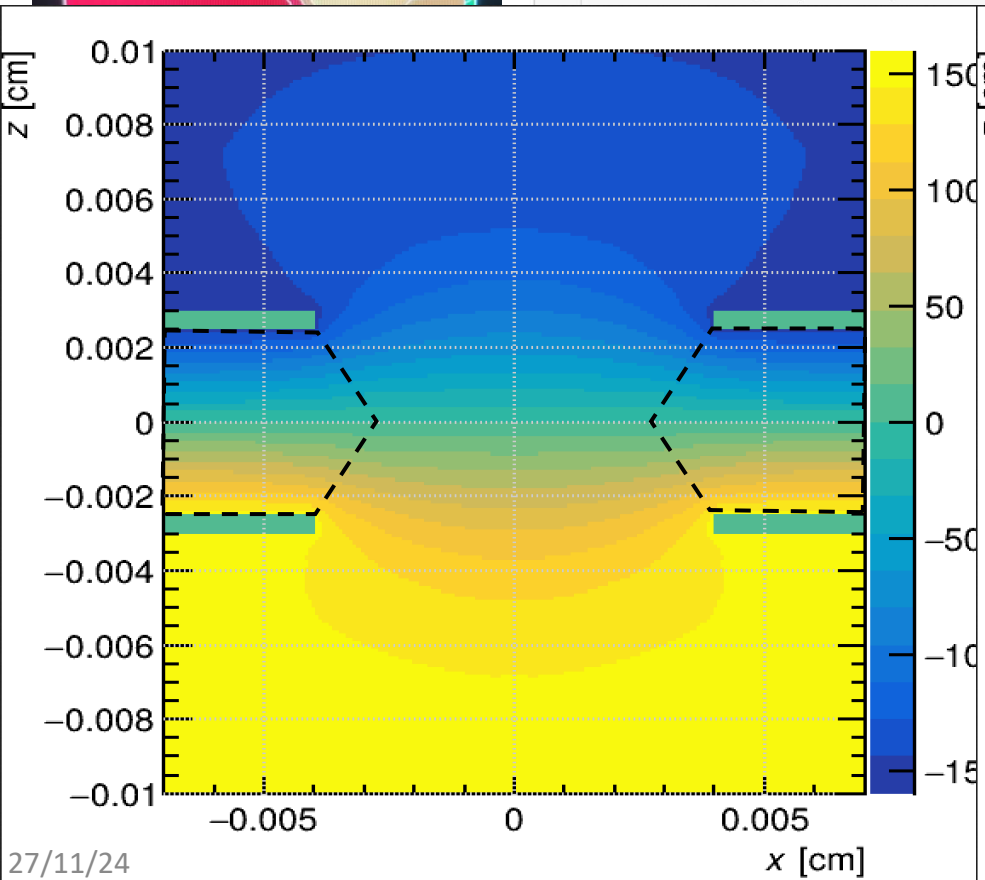
```python
compnt.SetGas(gas)                          # now link gas to this component
```

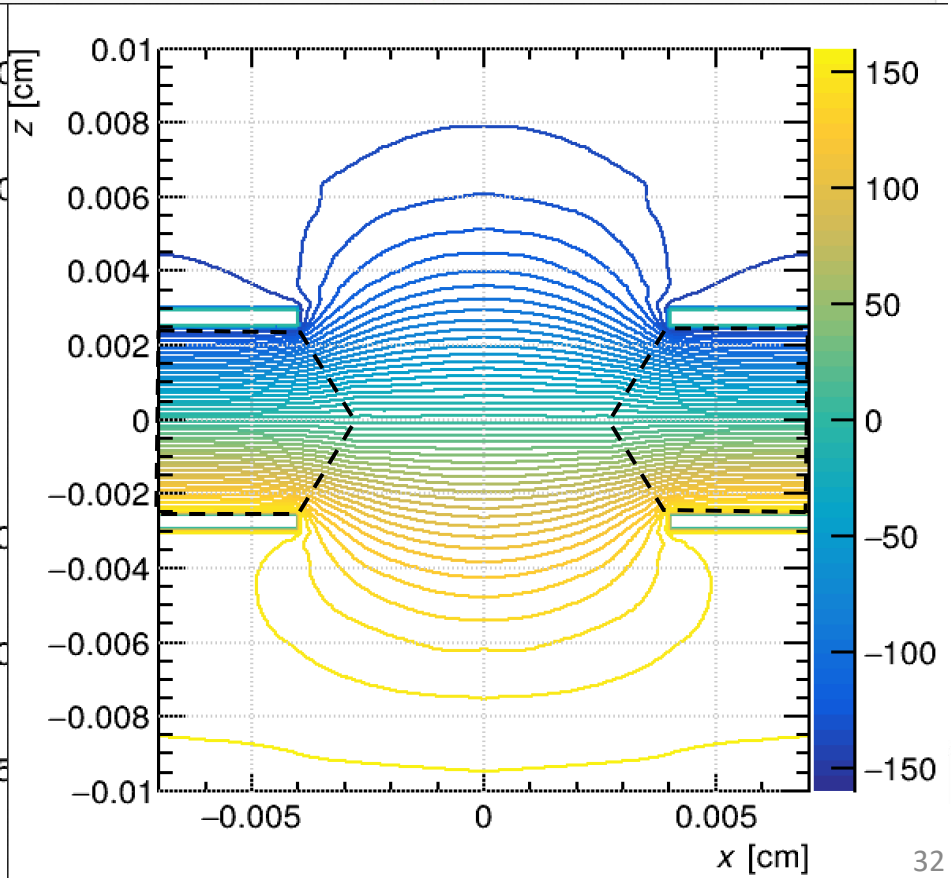
2. Electric Fields & Det Geometry



Create an helper object to plot the Electric Field

```
fieldView = ROOT.Garfield.ViewField(compnt) # and pass the component with E-field information as argument
cF1 = ROOT.TCanvas("cF1","",600,600)        # and create a ROOT canvas object to have all plot controlls
fieldView.SetCanvas(cF1)
fieldView.SetPlaneXZ()                      # lets view the XZ plane
fieldView.SetArea(-0.5*pitch, -0.01, 0.5*pitch, 0.01) # define the area where we'd like to see the Efield
fieldView.SetVoltageRange(-160,160)                   # use this to zoom in on the Efield
fieldView.GetCanvas().SetLeftMargin(0.16)             # plot aesthetics
fieldView.Plot("v", "colz")                           # "v" for voltage, "e" for electric field
```





3. Charged Particle Transport

Overview

Recall:

- We have a charged particle that produces primary ionization along its track
- We need to have this primary ionization moving (and in most cases to amplify it) to detect
  - Therefore we apply electrical fields in our detector
  - Now need to know how these particles move in the electric Field
- Lorentz: $F = q\,(E + v \times B)$
- Newton: $F = m\,\ddot{r}$
- Equation of motion: $r(x, v, t) = f(x, E, B)$
- But this is a macroscopic approach … we need to add **gas**

3. Charged Particle Transport

Intro: the mean free path in argon

- We know (e.g. from literature) that:
  - Diameter (Van der Waals) $r \approx 1.9\ 10^{-8}$ cm
  - Cross section of 1 atom: $\sigma \approx 1.5\ 10^{-16}$ cm^2
  - Atoms per volume: $n_0 \approx 2.7\ 10^{19}$ atoms/cm^3 *(Loschmidt)*
  - Average distance: $1/\sqrt[3]{n_0} \approx 3$ nm

$$n_0 = \frac{p_0}{k_B T_0}$$

- Mean free path for an electron ?
  - An electron hits all atoms of which the centre is less than a cross section σ radius from its path;
  - over a distance L, the electron hits $n_0\ \sigma\ L$ atoms;
  - mean free path = distance over which it hits 1 atom;
  - $\lambda_e = 1/(\sigma\ n_0) \approx 2.5\ \mu$m
  - much larger than
    - 3 nm distance between atoms in gas, and
    - 140-600 pm typical gas molecule diameters.

3. Charged Particle Transport

Intro: MPGDs and Mean free path

- Recall:
  - Mean free path of electrons in Ar: 2.5 µm,

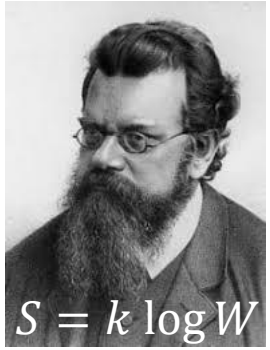
- Compare with:
  - Micromegas mesh pitch: 63.5 µm
  - GEM polyimide thickness: 50 µm
  - Micromegas wire thickness: 18 µm
  - GEM conductor thickness: 5 µm

- Hence:
  - mean free path approaches small structural elements;
  - such devices should be treated at a molecular level.

Slide © Rob Veenhof

3. Charged Particle Transport

Boltzmann equation



- The Boltzmann equation describes the evolution of the distribution function f (energy or velocity distribution) in 6D phase-space: f($\boldsymbol{x}, \boldsymbol{v}$) is function of position and velocity

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \nabla f + \frac{e}{m}(\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B}) \cdot \nabla f = 0$$

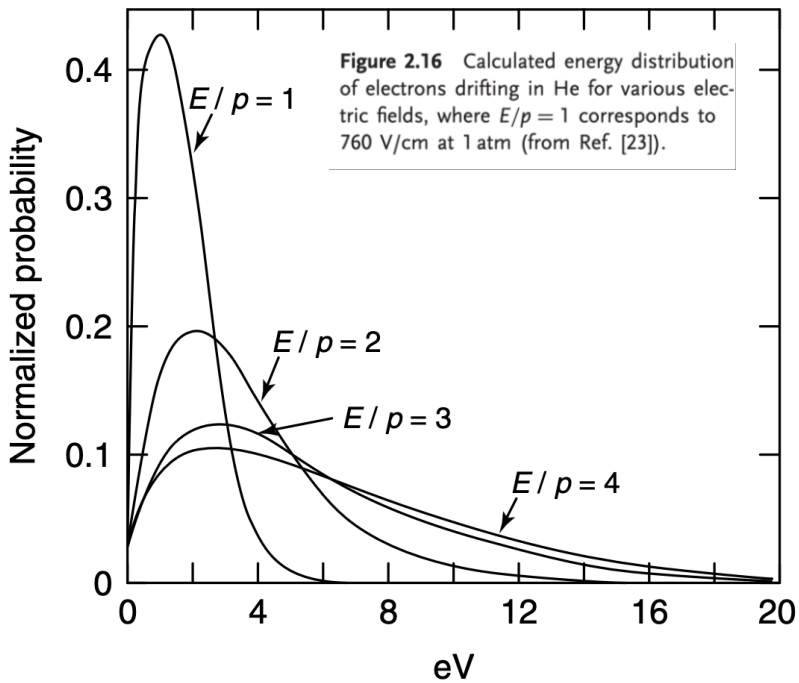


Figure 2.16 Calculated energy distribution of electrons drifting in He for various electric fields, where $E/p = 1$ corresponds to 760 V/cm at 1 atm (from Ref. [23]).

- 1935: solution by expansion of *f* with Legendre polynoms

$$f(x, v) = f_0(x, v) + \frac{\varepsilon}{v} f_1(x, v)$$

With f_0 the random distribution and f_1 the electron drift

- From 1960: numerical solutions with first computers
  - *Skullerud – Null-collision tech. 1968*
  - *Fraser & Mathieson 1986, E-fields*
  - *Steve Biagi 1998, E & B-fields*

3. Charged Particle Transport

Magboltz

- Magboltz solves transport equations in presence of E- and B-fields
- Program written in Fortran
  - http://cern.ch/magboltz
  - Integrated (through wrapper) in Garfield++
  - You need to update manually with new versions
- It uses electron-atom cross-sections
  - Internal database, published on LXCAT
  - Measurements -> cross section
- Precision of the cross-sections:
  - Total cross section: 1%
  - Ionisation 2%
  - Excitation 5-10%
- Magboltz calculates transport params
  - Townsend Coefficient
  - Attachment Coefficient
  - Drift velocity
  - Diffusion Tensor
  - Average electron energy

cern.ch/magboltz

CERN | Consult | Writeups | Magboltz

Magboltz - transport of electrons in gas mixtures

Responsible at CERN: Rob Veenhof
Manual Type: Source files, cross sections
Versions: 11.18
Author: Stephen Biagi
Reference: none

Created: 20 May 1995
Last Update: 31 Jan 2024
Verified: 31 Jan 2024
Valid until: further notice
Support Level: Normal

Magboltz

Magboltz solves the Boltzmann transport equations for electrons in gas mixtures under the influence of electric and magnetic fields.

Further information:

- LXCAT cross section compilation;
- How to use Magboltz

S.F. Biagi – Nucl. Instr Meth A 421 (1999)

ELSEVIER

Nuclear Instruments and Methods in Physics Research A 421 (1999) 234–240

NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH Section A

Monte Carlo simulation of electron drift and diffusion in counting gases under the influence of electric and magnetic fields

S.F. Biagi

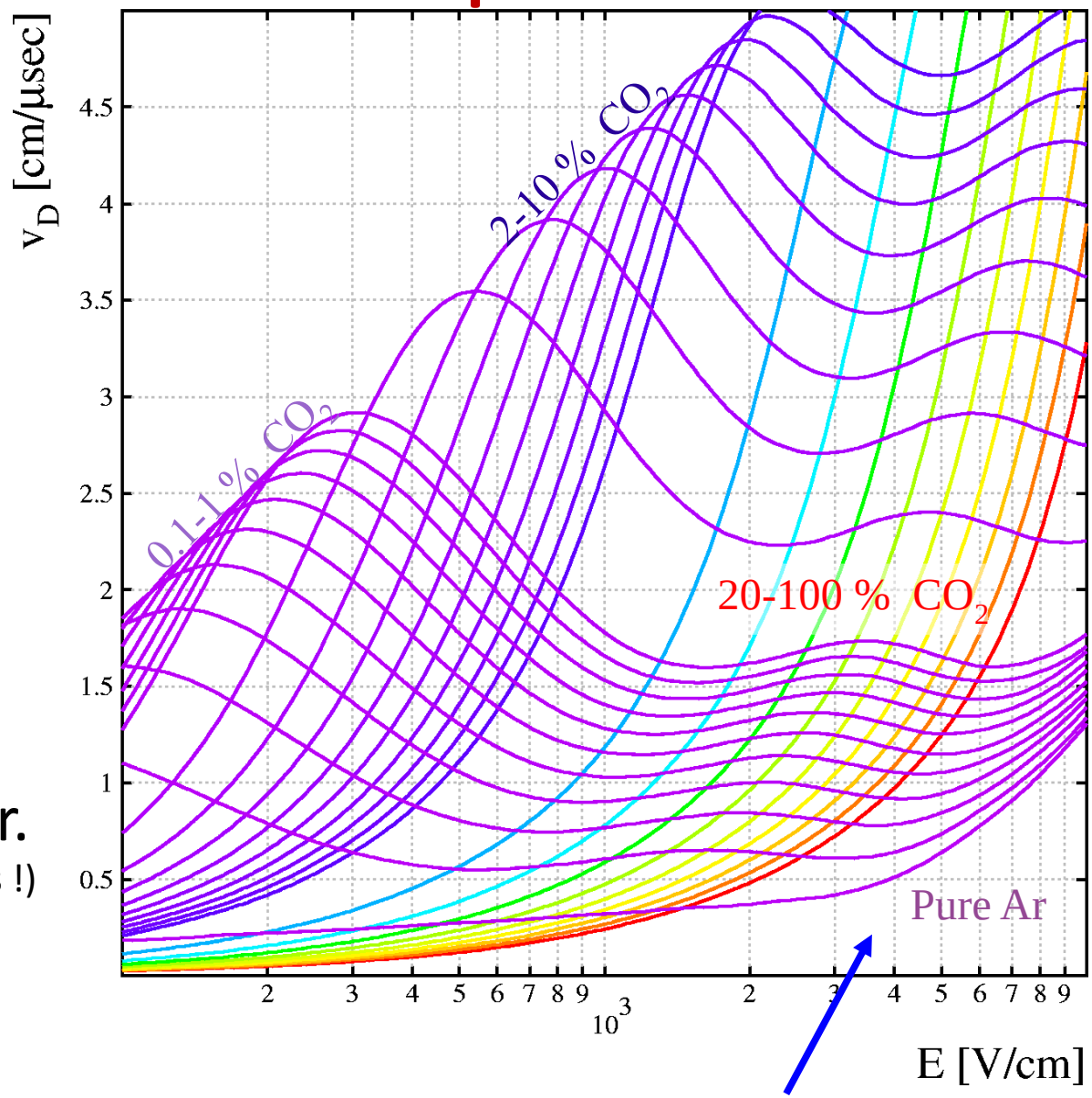
Department of Physics, Oliver Lodge Laboratory, The University of Liverpool, Liverpool L69 7ZE, UK

Received 20 July 1998; received in revised form 26 August 1998

3. Charged Particle Transport

Ar:CO_2

- CO_2 makes the gas faster, dramatically.
- Calculated by Magboltz for Ar/CO_2 at 3 bar.
  (Note where the arrow is !)

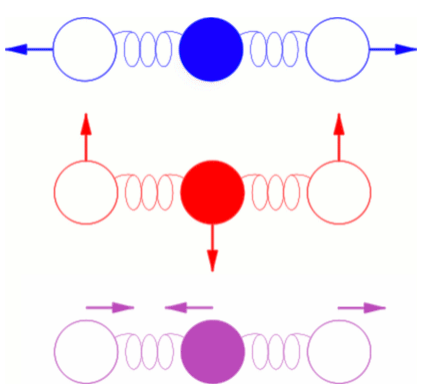


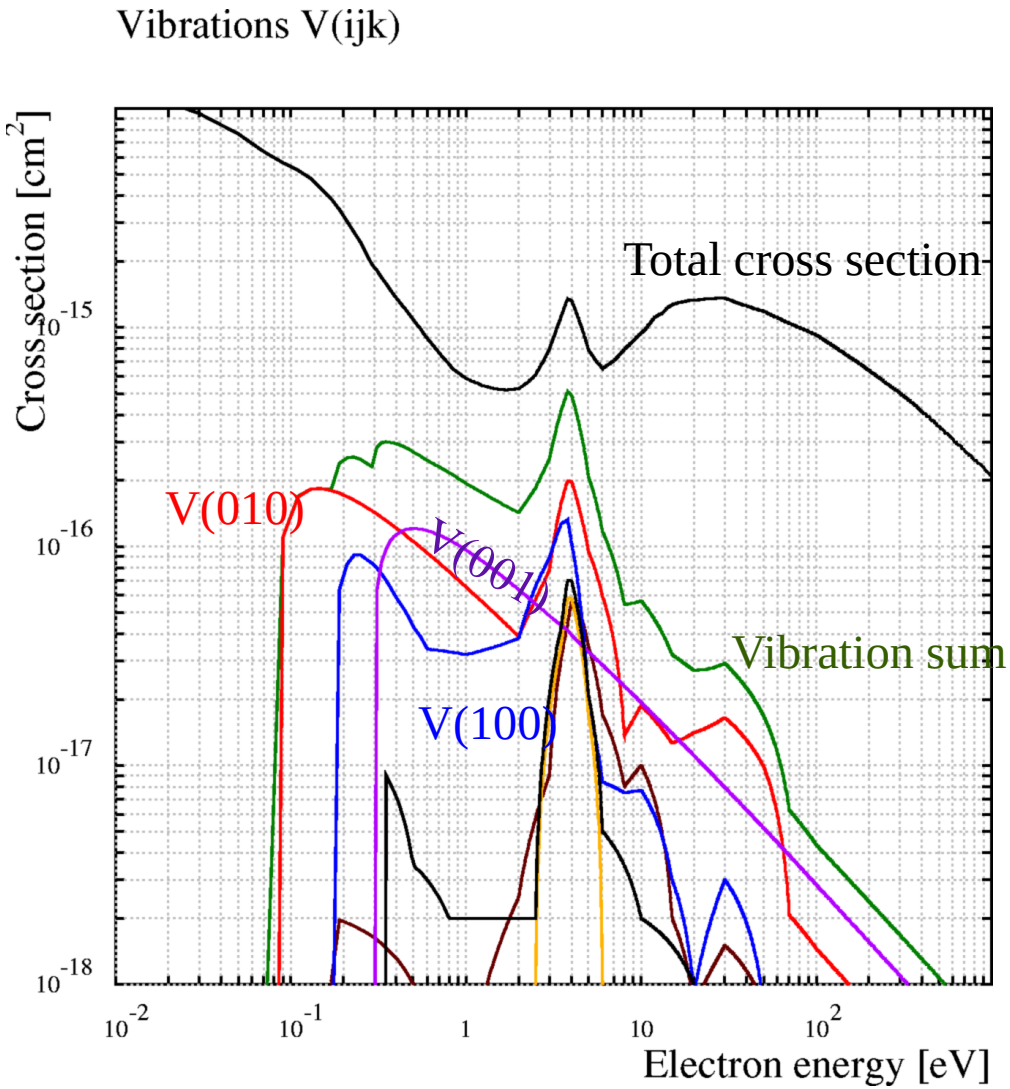
Slide © Rob Veenhof

3. Charged Particle Transport

CO_2 vibrational modes

- CO_2 is linear:
  - O – C – O
- Vibration modes are numbered V(*ijk*)
  - *i*: symmetric,
  - *j*: bending,
  - *k*: anti-symmetric.

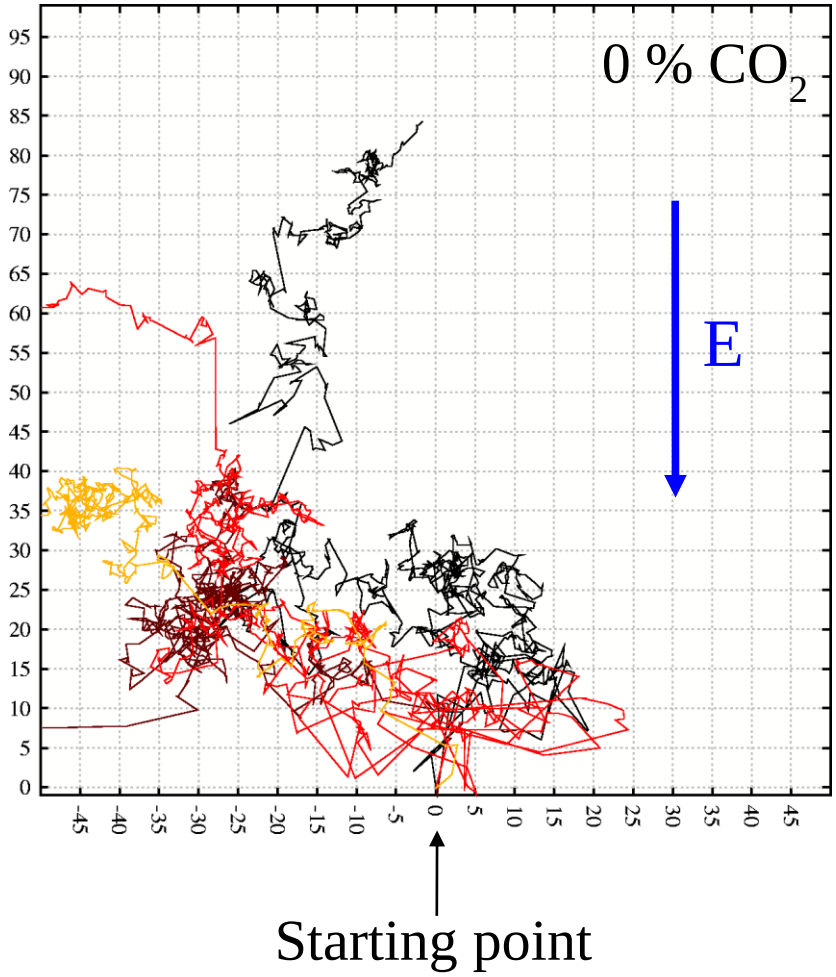


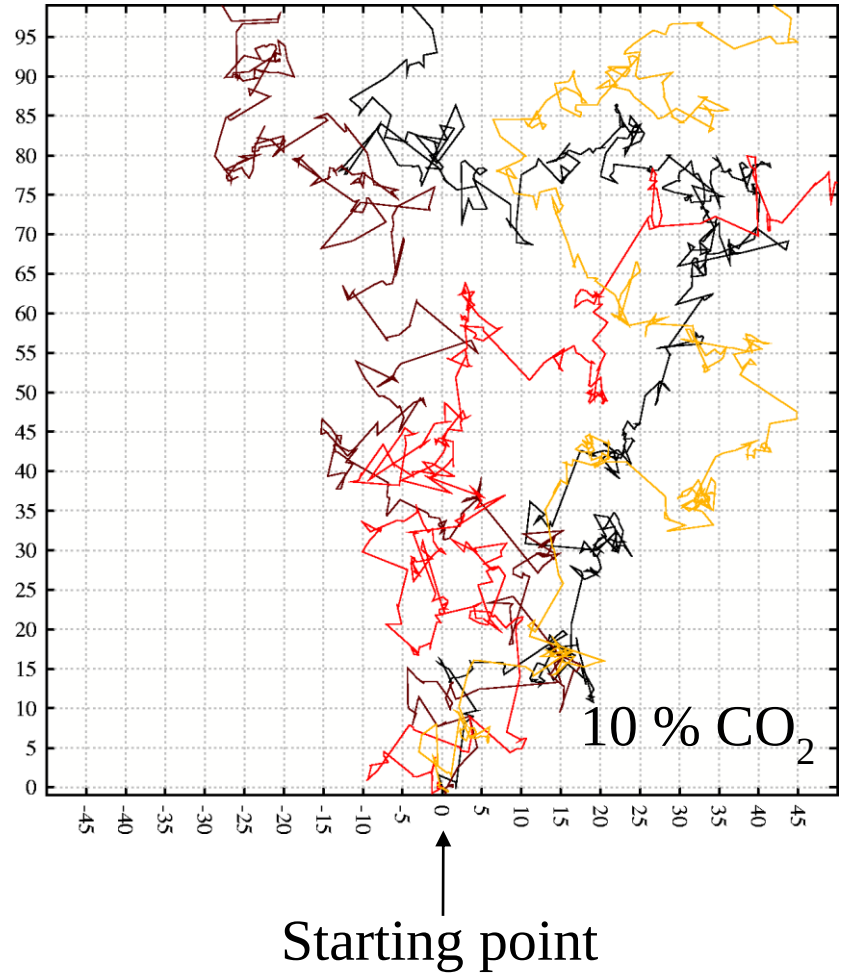


Slide © Rob Veenhof

3. Charged Particle Transport

Electrons in Ar:CO_2 mixtures at E = 1kV/cm

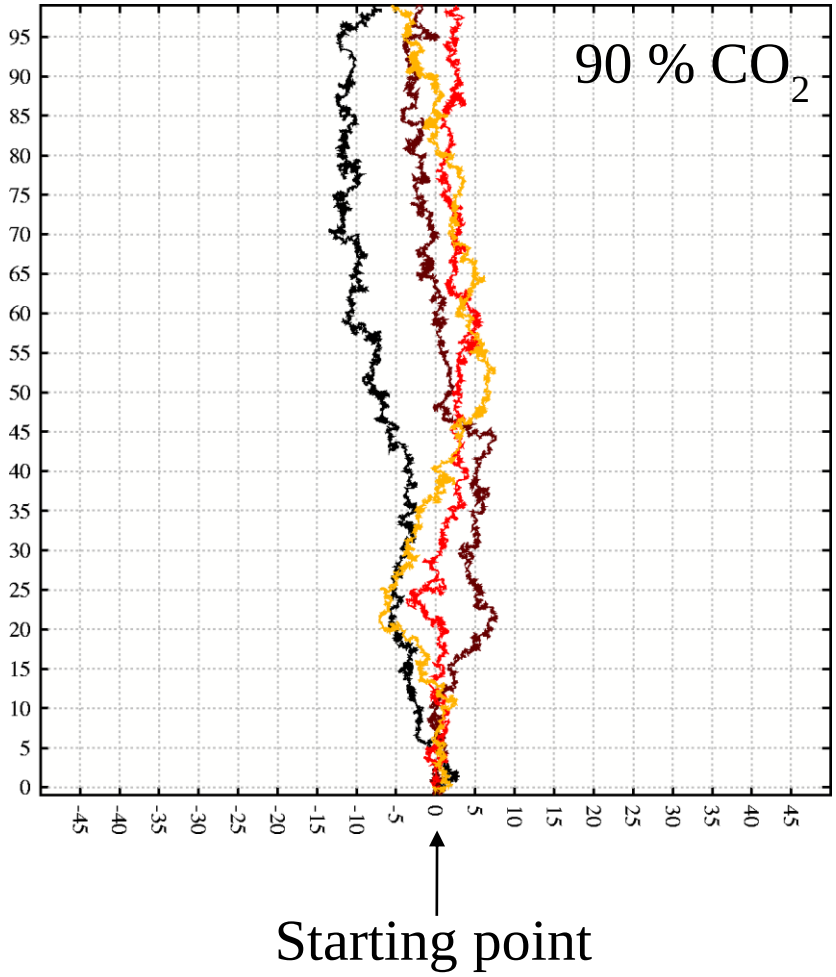


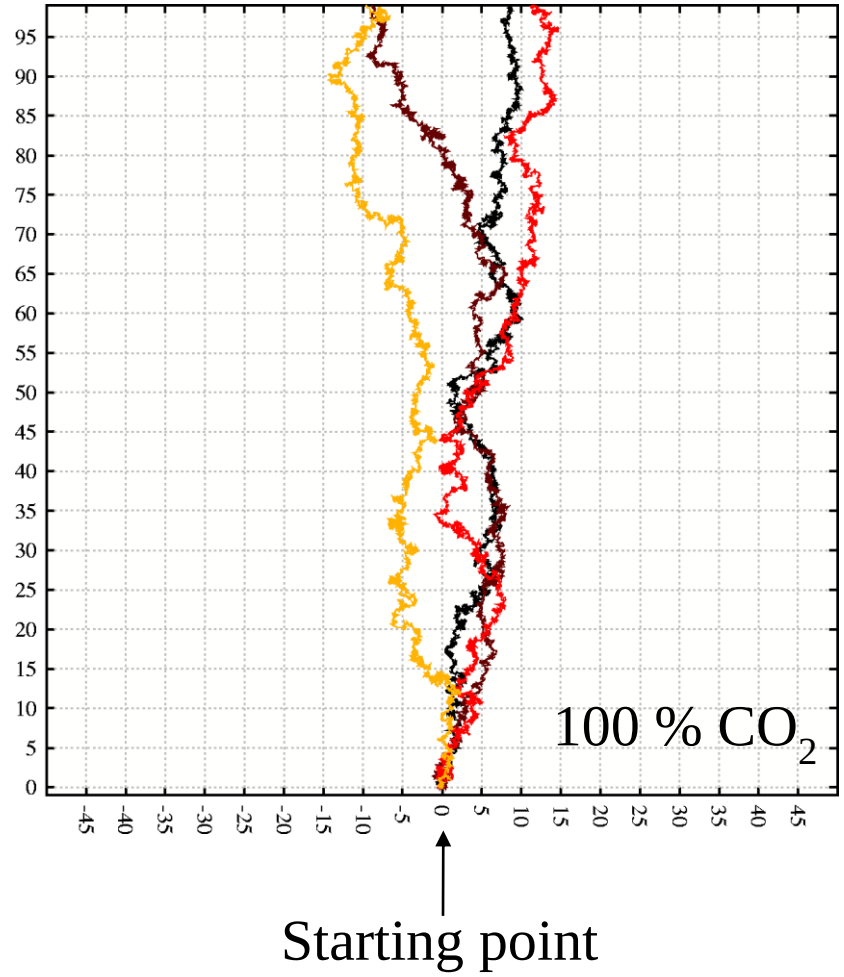


Slide © Rob Veenhof

3. Charged Particle Transport

Electrons in Ar:CO_2 mixtures at E = 1kV/cm



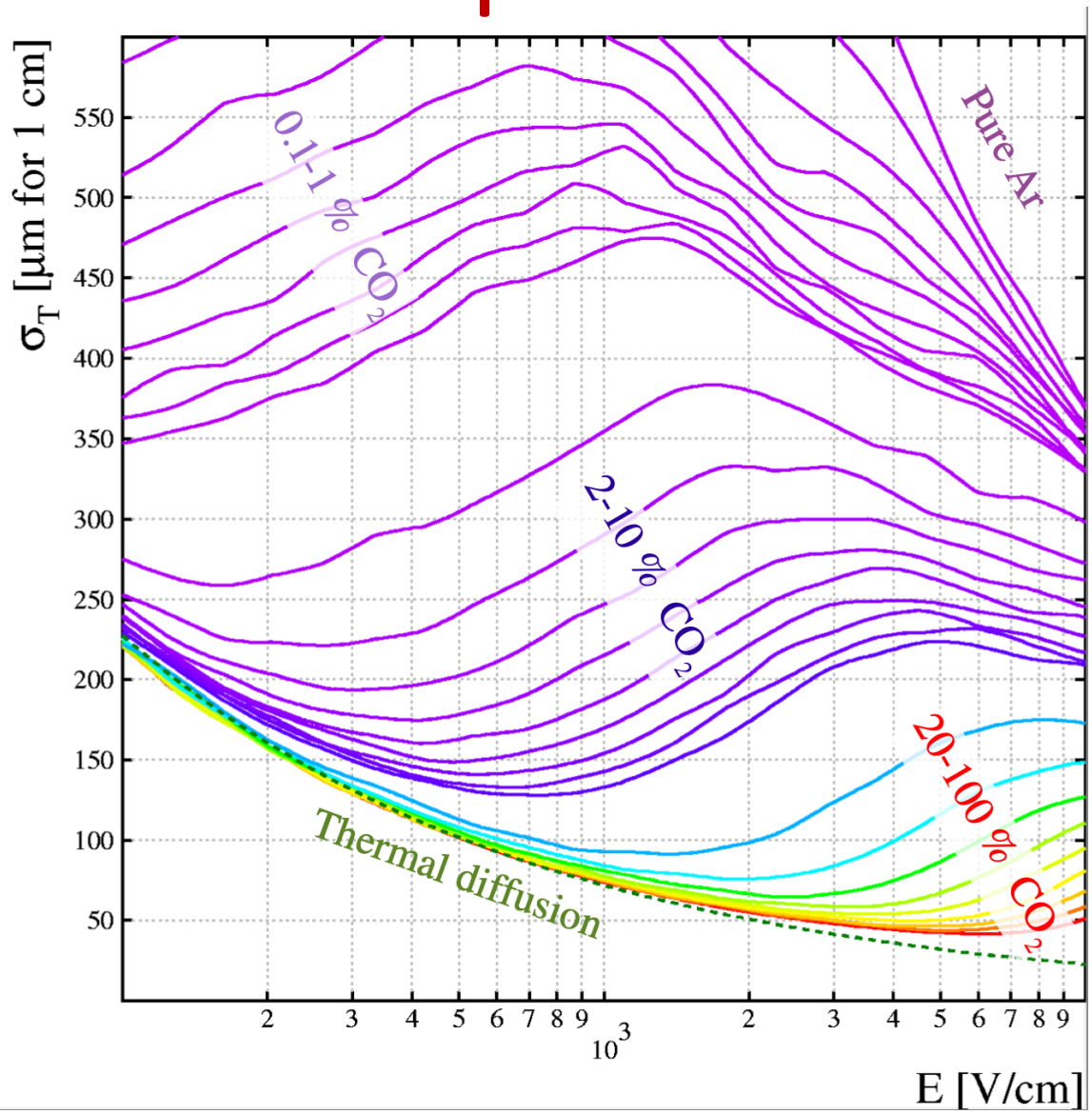


Slide © Rob Veenhof

3. Charged Particle Transport

Ar:CO_2

- Transverse diffusion is much reduced by CO_2.
- Calculated by Magboltz for Ar/CO_2 at 3 bar.



Slide © Rob Veenhof

3. Charged Particle Transport

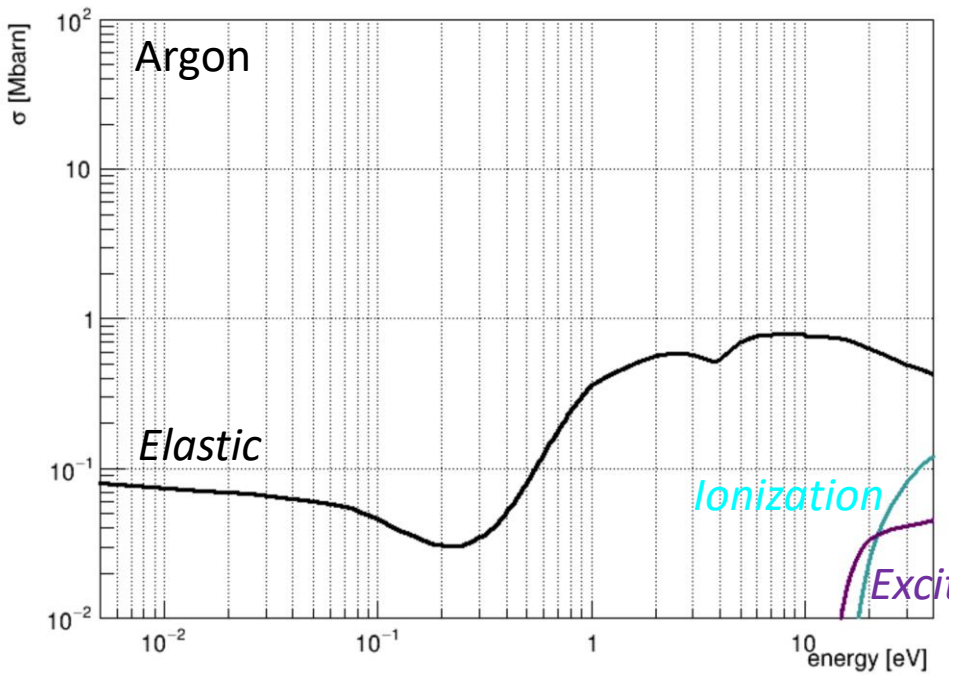
visualize Ar and CO_2 cross sections with Garfield++

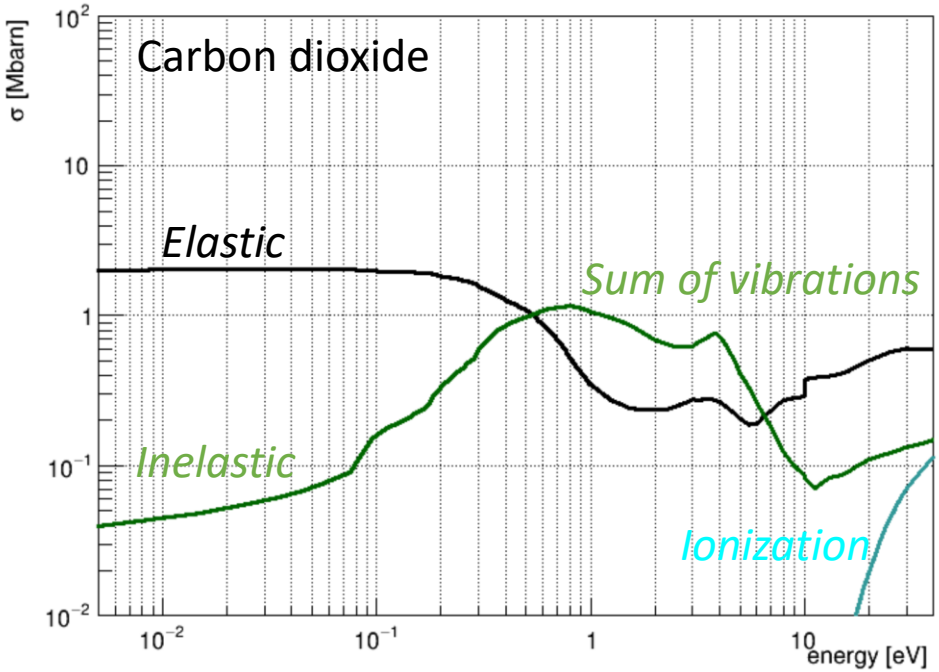
```
gas = ROOT.Garfield.MediumMagboltz("ar", 70., "co2", 30.)
```

```
MediumMagboltz::SetComposition: Ar/CO2 (70/30)
```

```
gas.PlotElectronCrossSections()
ROOT.gROOT.GetListOfCanvases().Print()
# Legend for the colours:
# Elastic     = kBlack
# Ionisation = kCyan - 2
# Attachment = kRed + 2
# Inelastic  = kGreen + 3
# Excitation = kMagenta + 3
```

Conversion factors:
1Mbarn = 10^{-18} cm^2



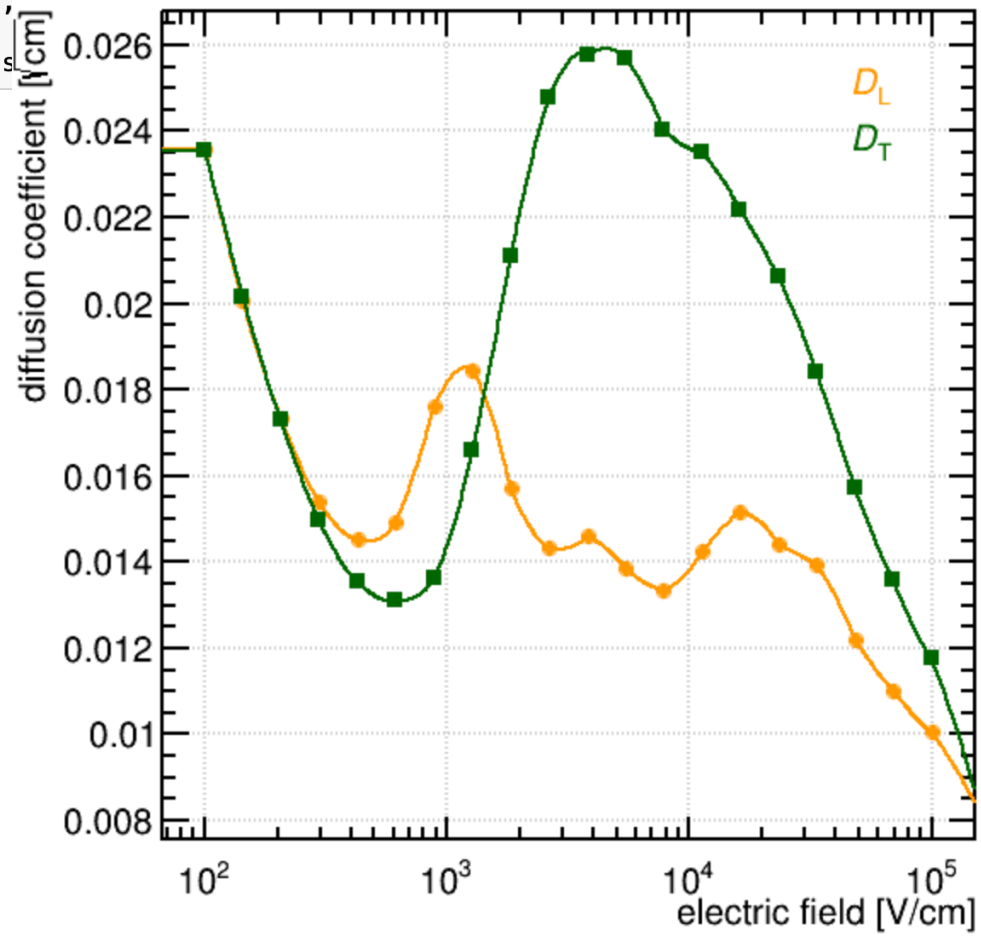


3. Charged Particle Transport

visualize Electron Velocity & Diffusion with Garfield++

```
gas = ROOT.Garfield.MediumMagboltz()
gas.LoadGasFile("ar_70_co2_30.gas")
```

```
cH = ROOT.TCanvas("cH","",
GasView.SetCanvas(cH)
GasView.PlotElectronDiffus
```

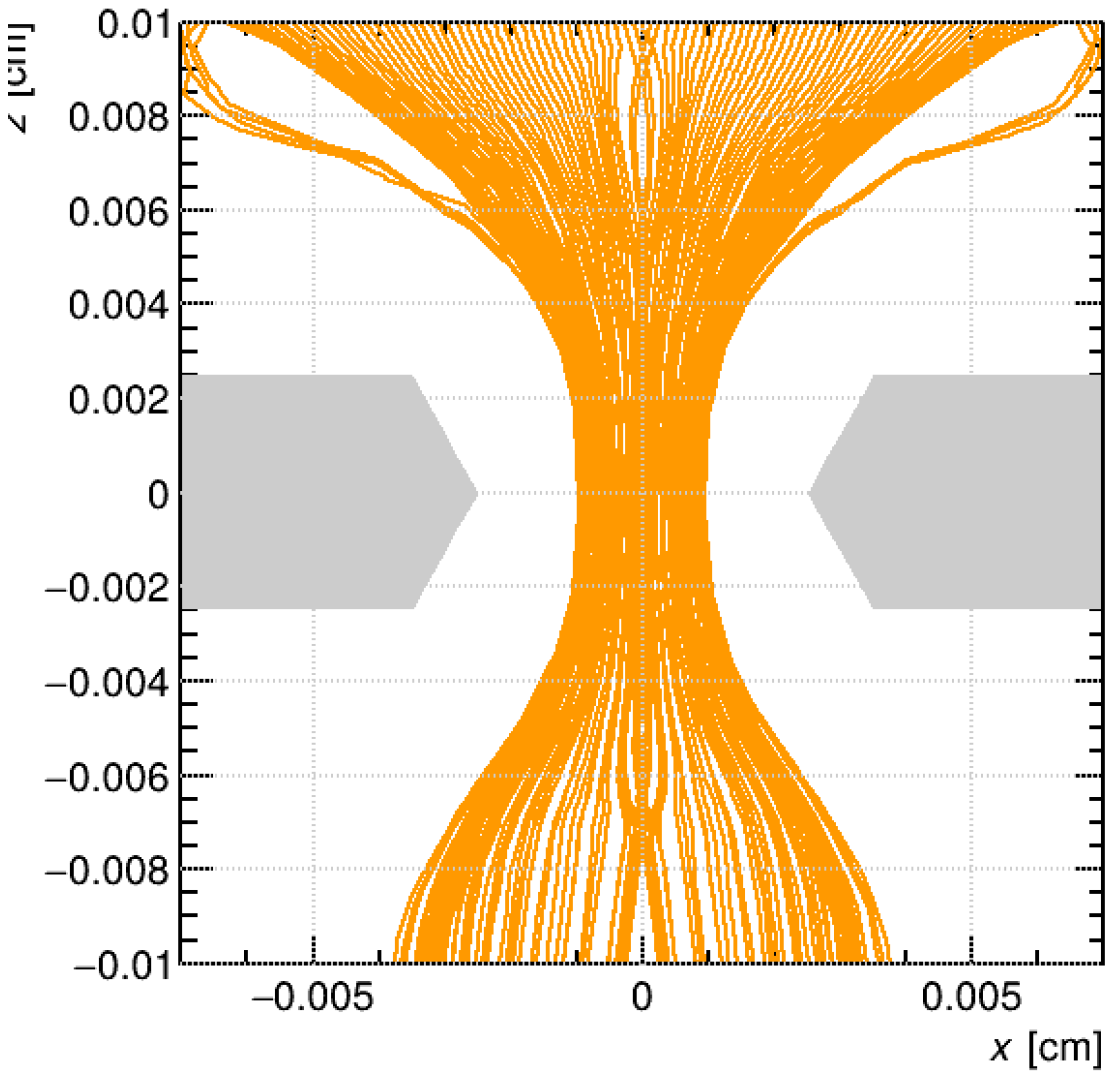


3. Charged Particle Transport

- Garfield++ has 3 different Transport Algorithms
  - Runge-Kutta-Fehlberg (RKF) Integration – numerical method to solve ODEs
  - Monte-Carlo
  - Microscopic Tracking
- First 2 methods calculate the electron trajectory, using as input the Transport Parameters (α,η, v_d, D_L,D_T)
  - These parameters need to be provided as a gasfile (map/table), in function of E-field, B-field and angle between E & B field
  - This gasfile can be calcuted in Garfield++ using Magboltz
- 3rd method uses the electron-Atom scattering cross sections
  - Cross-sections taken from Magboltz and hard-coded in Garfield
  - Need manual update each time a gas is updated by Steve
- Different methods have different level of precision, resources,..
  - Wire-based detectors: OK to use RKF integration
  - Micro-Pattern detectors: need to use Microscopic Tracking
  - Parallel plate: use Microscopic Tracking up to O(100) electrons

3. Charged Particle Transport

Illustration of Runge Kutta Fehlberg




```
ld/Data/IonMobility_Ar+_Ar.txt")
" K")
")
```

This time we do not define the gas ourself on the spot, but we load a gasfile

- Temperature and Pressure predefined
- Ar:CO_2 80:20 – varying E-field, B = 0T

This is the core … we define an object DriftLineRKF and we pass the sensor

Now we define some helper objects

```
0.01)
```

- Visualize the geometry (ViewFEMesh)
- Visualize the trajectories (ViewDrift)

we define some starting points
Equally distribute 101 points in [–pitch/2 +pitch/2]

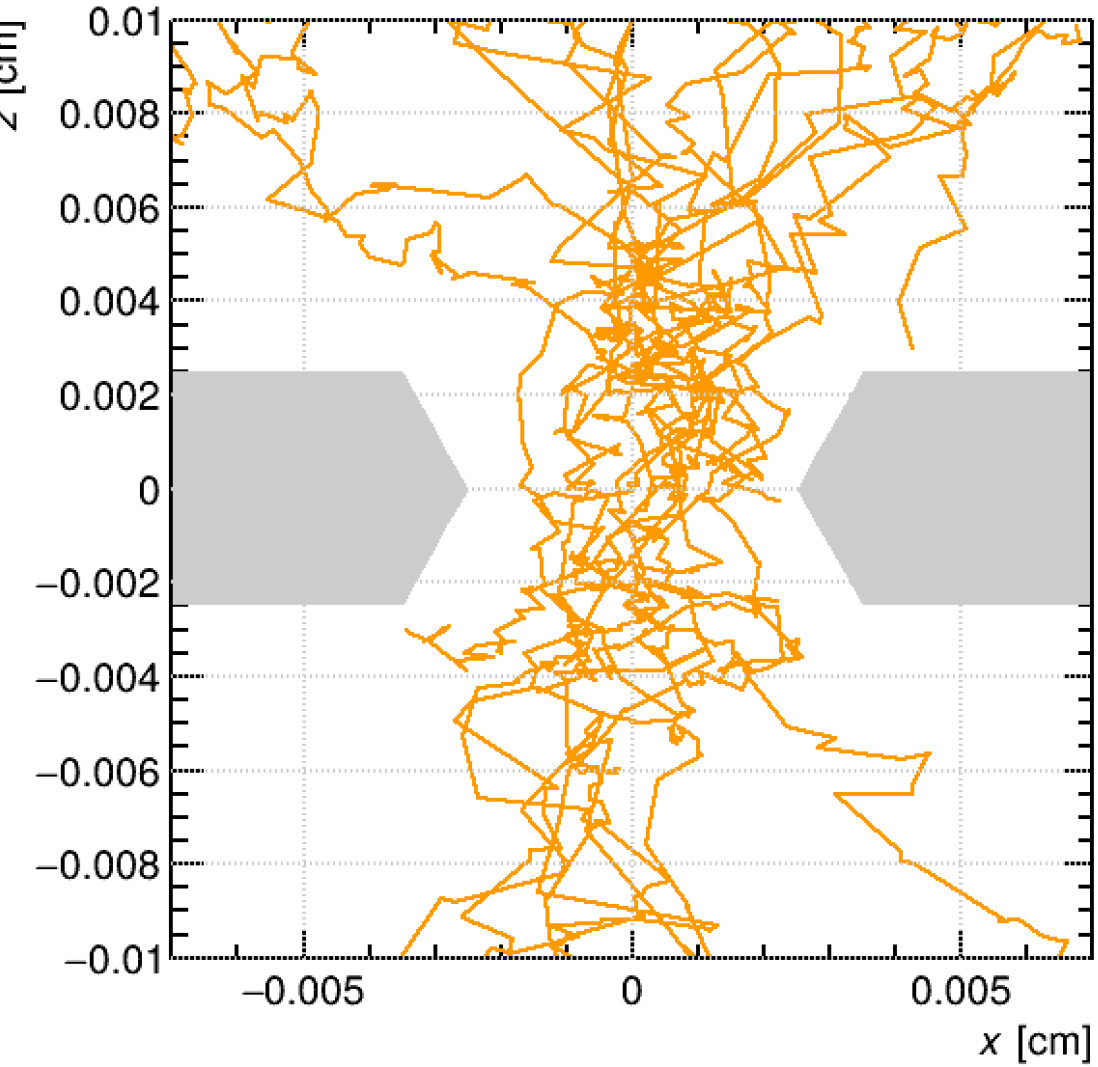
```
Electron(x0,y0,z0,t0)
```

We launch an electron at t0=0ns for each starting point x

```
bine Geometry view & Trajectory view & plot
```

3. Charged Particle Transport

Illustration of Monte Carlo Integration




```
eld/Data/IonMobility_Ar+_Ar.txt")
 " K")
r")
```

This time we do not define the gas ourself on the spot, but we load a gasfile

- Temperature and Pressure predefined
- Ar:CO_2 80:20 – varying E-field, B = 0T

This is the core … we define an object AvalancheMC and we pass the sensor

```
, 0.01)
```

Now we define some helper objects

- Visualize the geometry (ViewFEMesh)
- Visualize the trajectories (ViewDrift)

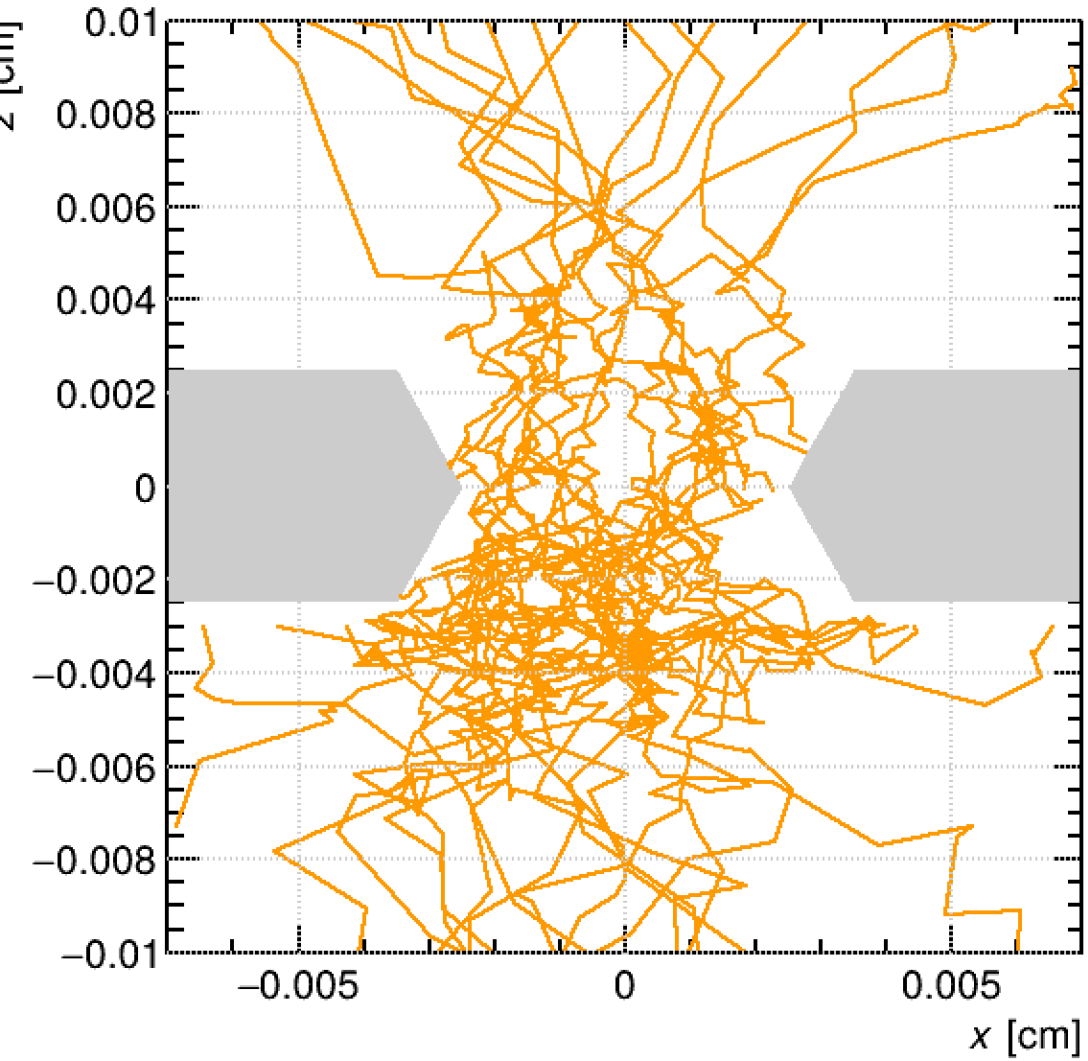
```
tElectron(x0,y0,z0,t0)
```

We launch an electron at t0=0ns for each starting point x

bine Geometry view & Trajectory view & plot

3. Charged Particle Transport

Illustration of Microscopic Tracking




```
arfield/Data/IonMobility_Ar+_Ar.txt")
```

For Microscopic Tracking we need access to the electron-Atom cross sections

And we do **not** need the Transport Params

```
arfield/Data/IonMobility_Ar+_Ar.txt
```

```
sor)
```

This is the core … we define an object AvalancheMicroscopic & pass the sensor

```
0.01)
```

Now we define some helper objects

- Visualize the geometry (ViewFEMesh)
- Visualize the trajectories (ViewDrift)

We launch an electron

- at position x0, y0, z0 and t0=0ns
- With energy e0, and direction dx0, dy0, dz0

```
,0.,0.) # AvalancheElectron(x0,y0,z0,t0,e0,dx0,dy0,dz0)
```

nbine Geometry view & Trajectory view & plot

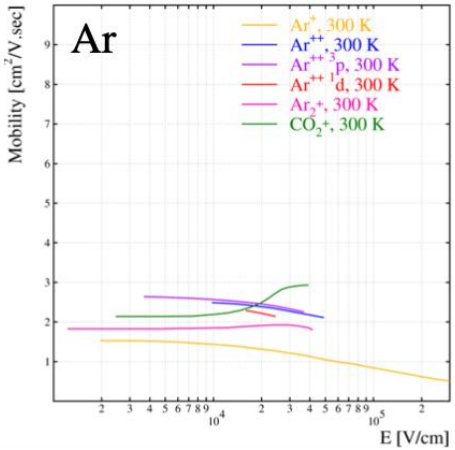
3. Charged Particle Transport

Ion transport

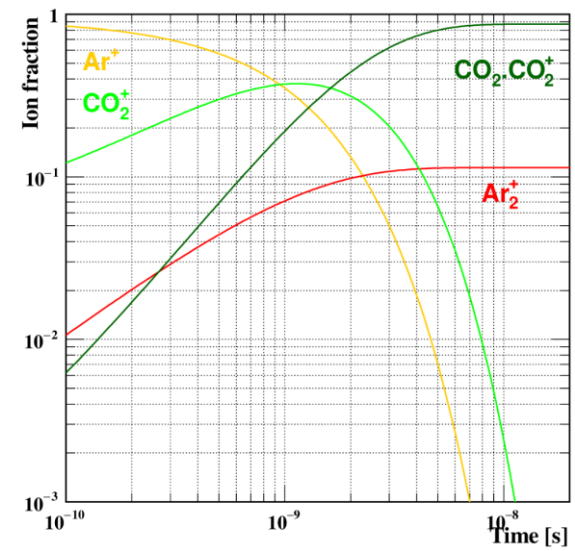
- Ions are produced in primary ionization and in avalanches, in same quantities
- GEMs get their signal mostly from moving electrons
- Wire chambers, Micromegas, uRWELL get most of their signal from the motion of the ions
- Therefore important to know:
  - Which ions are moving?
  - How fast are they moving?
  - Do they diffuse?

3. Charged Particle Transport

Ion transport



- Ions are > 1000 times more massive than electrons
  - This reflect in their drift velocity that is ~ 1000 x slower
- We define the Ion mobility as $\mu = vd/E$
- Ion Mobilities are well known for the noble gases
- In Ar:CO_2: Ar+ dominates in the ionization (but CO_2+ dominates in Ne:CO_2)
- However:
  - Formation of dimers: $Ar^+ + Ar \rightarrow Ar_2^+$
  - Charge exchange: $Ar^+ + CO_2 \rightarrow Ar + CO_2^+$
  - Cluster formation: $CO_2^+ + 2\,CO_2 \rightarrow CO_2^+ \circ CO_2 + CO_2$
    - *Formation time: 7ps – decay time 5ns … long-lived cluster*
- *This is not fully implemented in the simulation*
  - *Instead of $CO_2^+ \circ (CO_2)_n$ mobility in Ar:CO_2 mix, we use Ar^+ in Ar*
  - *Correct within 10-15% … but for other mixtures this is far worse …*
  - *Simulation provides Ar+ in Ar, Ne+ in Ne, He+ in He and few others …*
  - *Lot's of room for improvement, especially if correct signal shape is important*

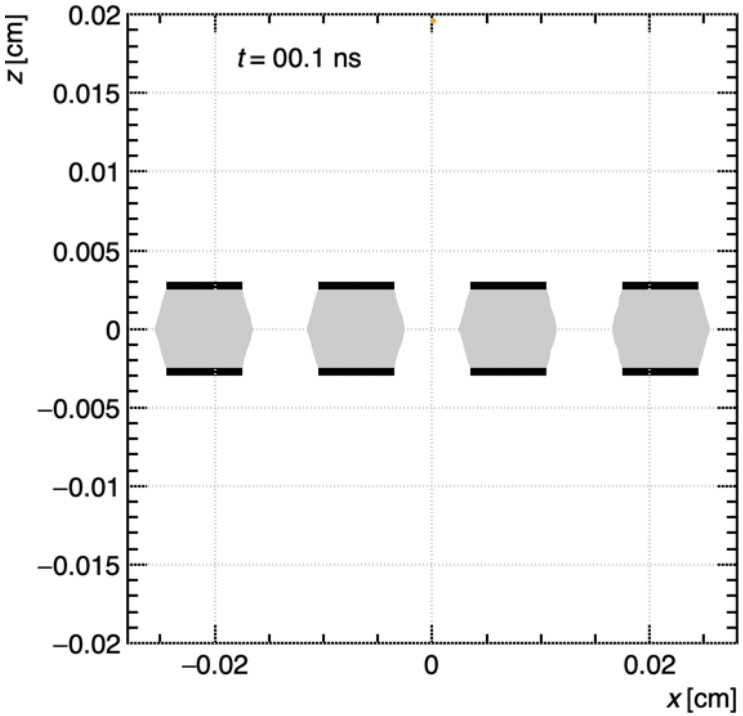


Figures © Rob Veenhof

4. Charge Amplification

- Townsend coeff α: probability per unit length that an electron creates an additional electron
  - $dn(x) = n(x)\alpha(x)dx$ => $n(x) = n(0)\exp(\int_0^x \alpha(x')dx')$

- Once field is high enough electron picks up energy > IP
  - Electron can ionize an atom in collision
  - Liberated electrons sense the strong field and can ionize further
- Implemented in all Electron Transport classes
  - Runge-Kutta-Fehlberg (e-, ion+)
  - Avalanche MC (e-)
  - Microscopic Tracking (e-)



Second Lecture

see you Monday!

Questions?

Stupid questions do not exist

References

References & Acknowledgements

- References:
- RD51 Simulation School 2011 https://indico.cern.ch/event/110634/
- RD51 Open Lectures 2017 https://indico.cern.ch/event/676702/
- RD51 Open Lectures 2021 https://indico.cern.ch/event/911950/
- Acknowledgements
  - *To Rob, who taught many of us simulation*
  - *And all – major and minor contributors- who have developed Garfield++ code and who have developed the exercises over the past 14++ years*

Third Lecture

hands-on exercises

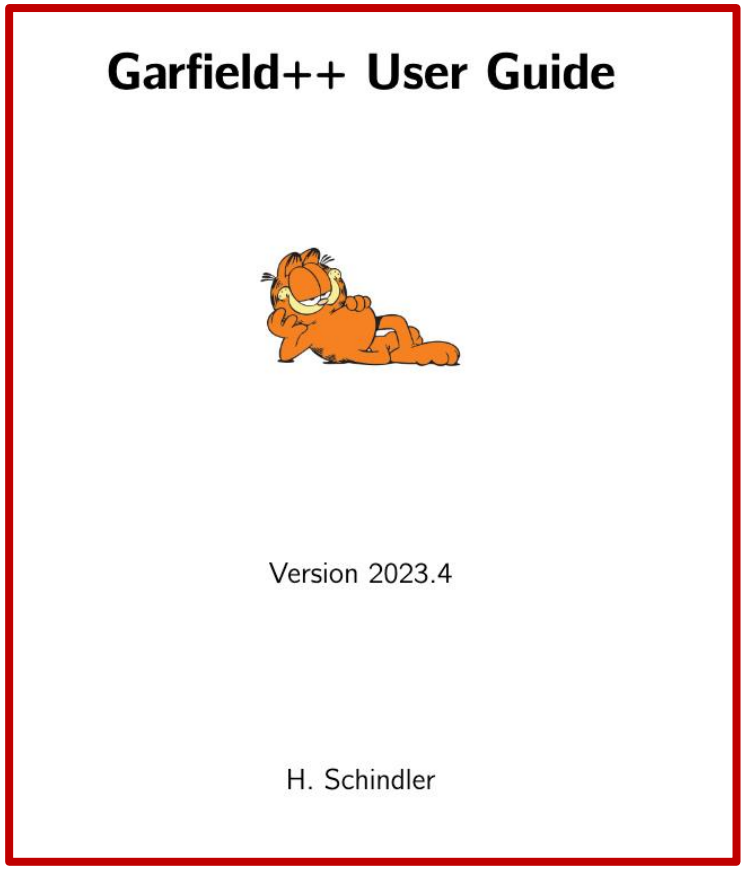
Hands-On Exercises

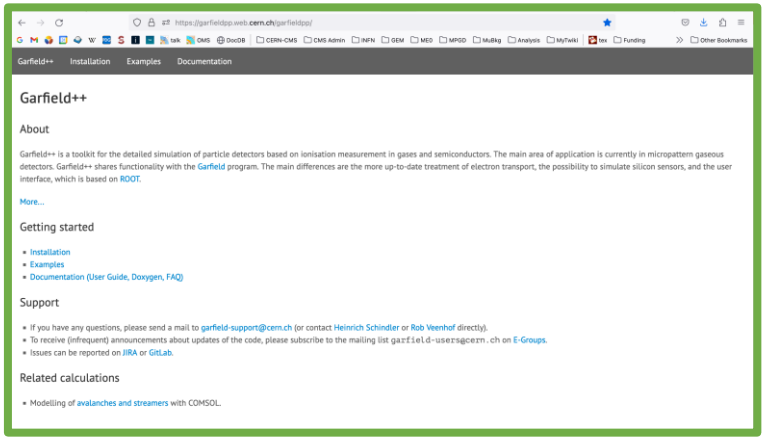
- We have prepared a Linux Virtual Machine
  - Xubuntu 24.04 LTS – ROOT and Garfield++ preinstalled
  - You can download the VM at the following location: https://cernbox.cern.ch/s/2GwgQCBQ0TSMnMc
  - You need to download Oracle Virtual Box to open it
  - Username: “student” – Password: “password”
- The Linux VM contains also the Hands-On exercises
  - /home/student/DRD1-School-2024

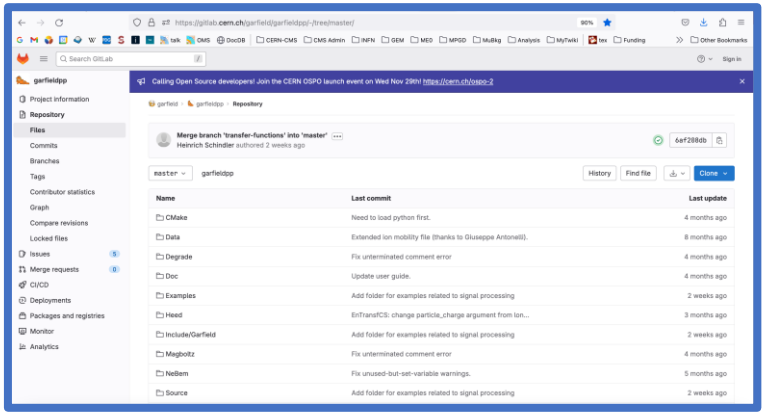
Hands-On Exercises

- Introductory Simulation Lab
  - 3-5 Simple exercises, dedicated to:
    - Detector setup, electric field
    - Primary Ionization
    - Avalanche Multiplication
    - Signal Induction
    - Influence of resistivity on signal induction
- Advanced Simulation Lab
  - 1-2 big (full) exercises – *in group – with strong help …*
    - Signal in Resistive Place Chambers
    - Signal in Resistive Micromegas

Your Friends:







Your best friends: the Holy Trinity: The Manual, The Source Code, The Examples

- *Maybe also your tutors - we will do our best* ☺
- https://garfieldpp.web.cern.ch/garfieldpp/documentation/UserGuide.pdf
- https://gitlab.cern.ch/garfield/garfieldpp/-/tree/master/
- https://garfieldpp.web.cern.ch/garfieldpp/documentation/

For Master / PhD students

with lxplus account

- https://garfieldpp.web.cern.ch/garfieldpp/getting-started/
- Instructions for bash shell:
  - Login (lxplus 8)
    - `ssh <username>@lxplus8.cern.ch`
  - Pickup latest (nightly) built of Garfield++
    - `source /cvmfs/sft.cern.ch/lcg/views/dev3/latest/x86_64-el8-gcc11-opt/setup.sh`
    - `source /cvmfs/sft.cern.ch/lcg/views/dev3/latest/x86_64-el8-gcc11-opt/share/Garfield/setupGarfield.sh`
    - `export GARFIELD_HOME=$GARFIELD_INSTALL`
  - Pick-up an example C++ file, compile and execute it
    - `cp -r $GARFIELD_HOME/Examples/Gem .`
    - `mkdir Gem/build; cd Gem/build`
    - `cmake ..`
    - `make`
    - `./gem`