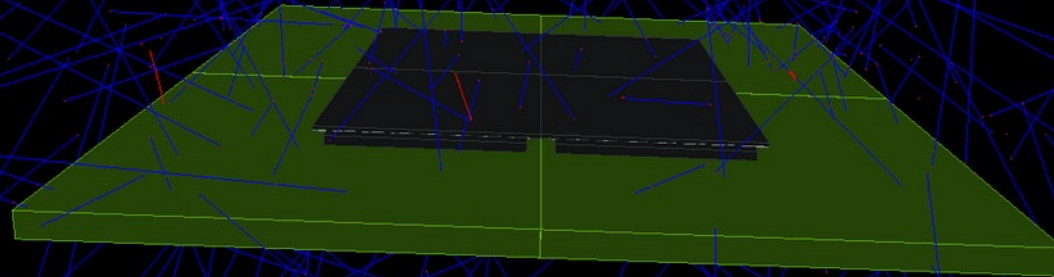




cern.ch/allpix-squared

The Allpix Squared Framework

Silicon Detector Monte Carlo Simulations for Particle Physics and Beyond



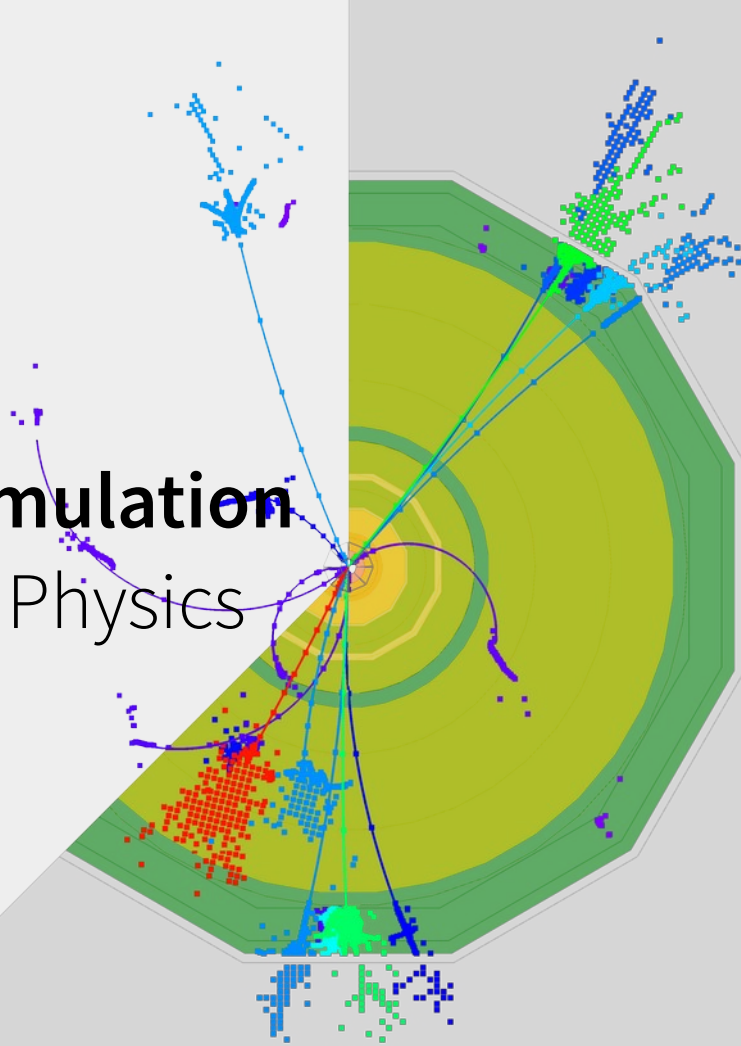
Simon Spannagel, Paul Schütze – DESY

ESA / ESO Detector Modelling Workshop

16 June 2021

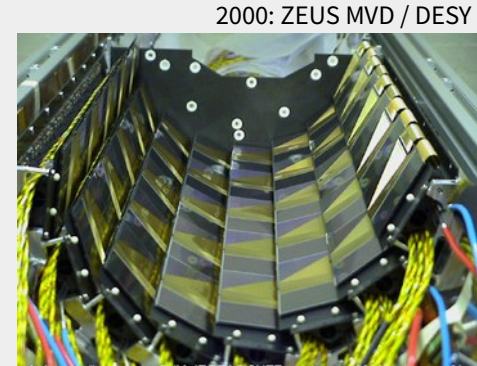
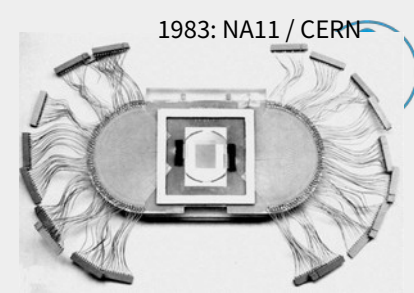
Silicon Detectors & Simulation

in High-Energy Particle Physics



Silicon Detectors in Particle Physics

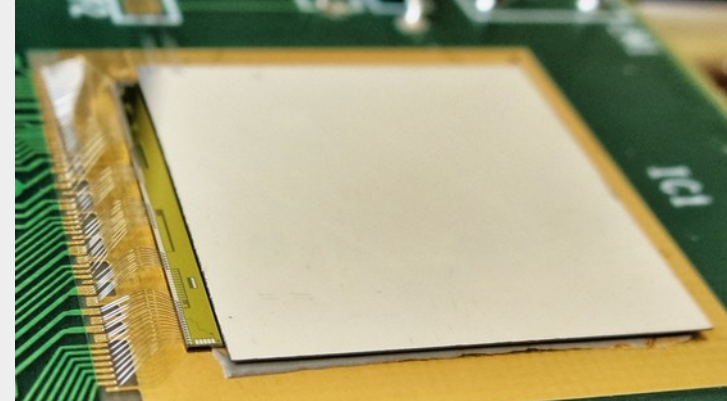
- Silicon detectors vital for many measurements
 - Fine segmentation, fast readout: high track multiplicities
 - Precise position measurement: momentum determination
collision point (vertexing)
particle identification (flavor tagging)
- Instrumental in discovery of Higgs boson at LHC
 - Tracking detectors: strips, 200 m² silicon, 70M channels
 - Vertex detectors: pixels, 1 m² silicon, 140M channels
- Detector R&D underway for
 - Upgrade of HL-LHC: more radiation damage resilience
 - Future colliders: *faster, higher, better*



Silicon Detectors in Particle Physics

Demands on detectors are high:

- Very high particle flux, 10s MHz / cm²
- Maximum resolution, minimum (scattering-) mass
- Very high granularity for high particle rates, fast readout, minimal dead time (few ns)
- “Smart” detectors (zero suppression, clustering, on-chip processing, fast data links)



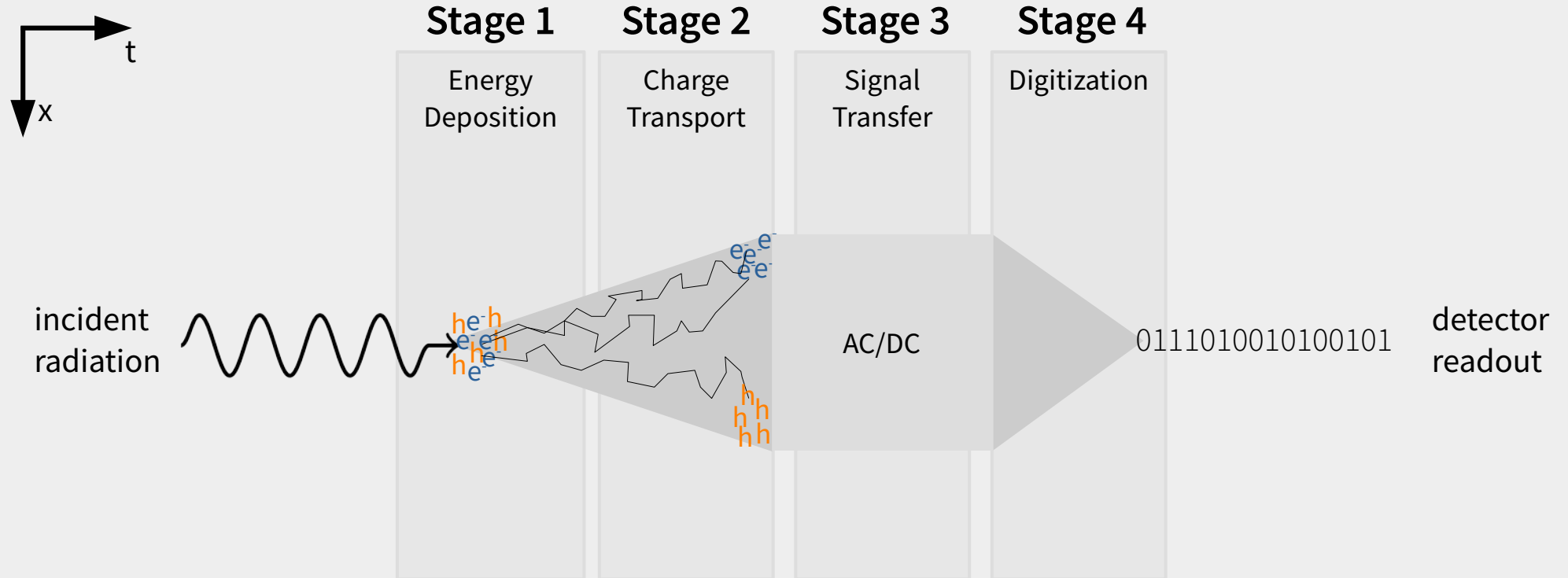
100 μm Timepix with 100 μm Sensor

Many different technologies used for different purposes:

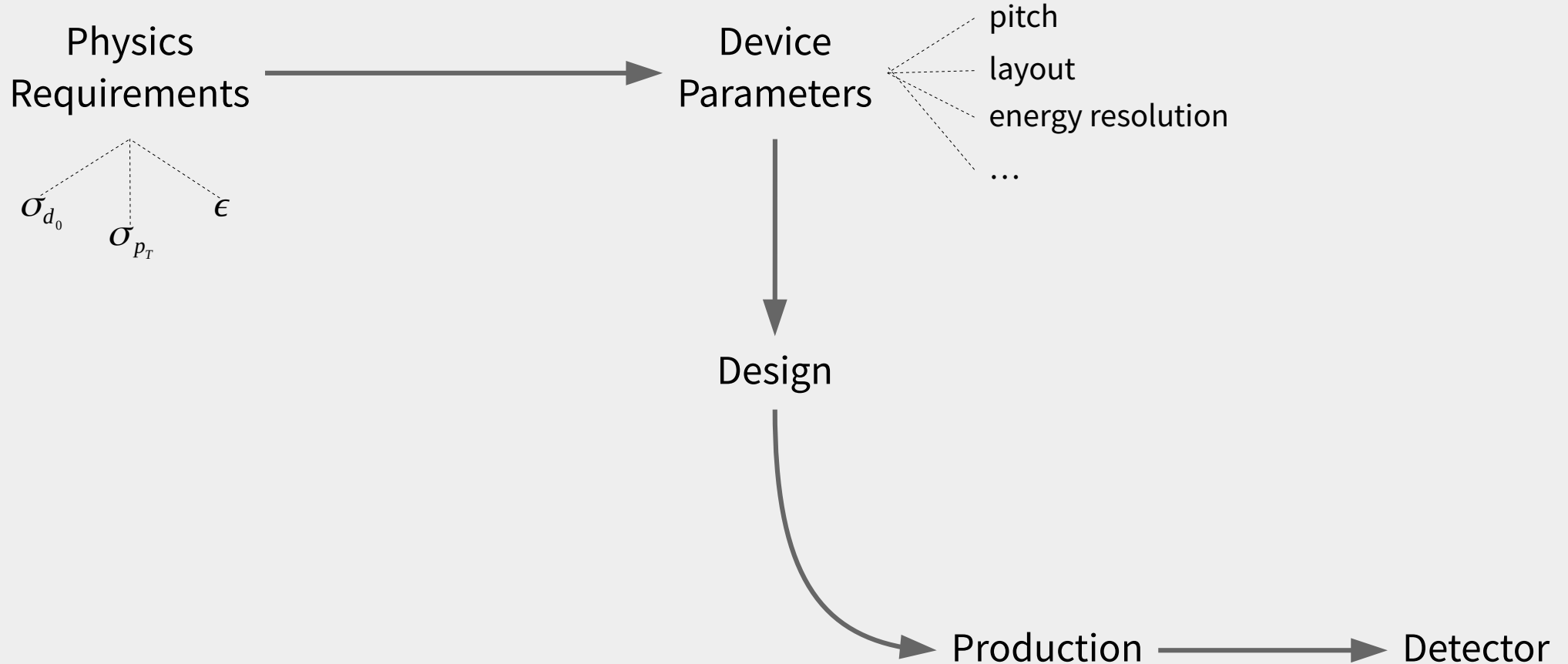
hybrid – dedicated sensor + mixed-mode CMOS, monolithic CMOS imaging, LGADs, 3D sensors, ...

Simulations for thoroughly understanding detector performance in realistic conditions

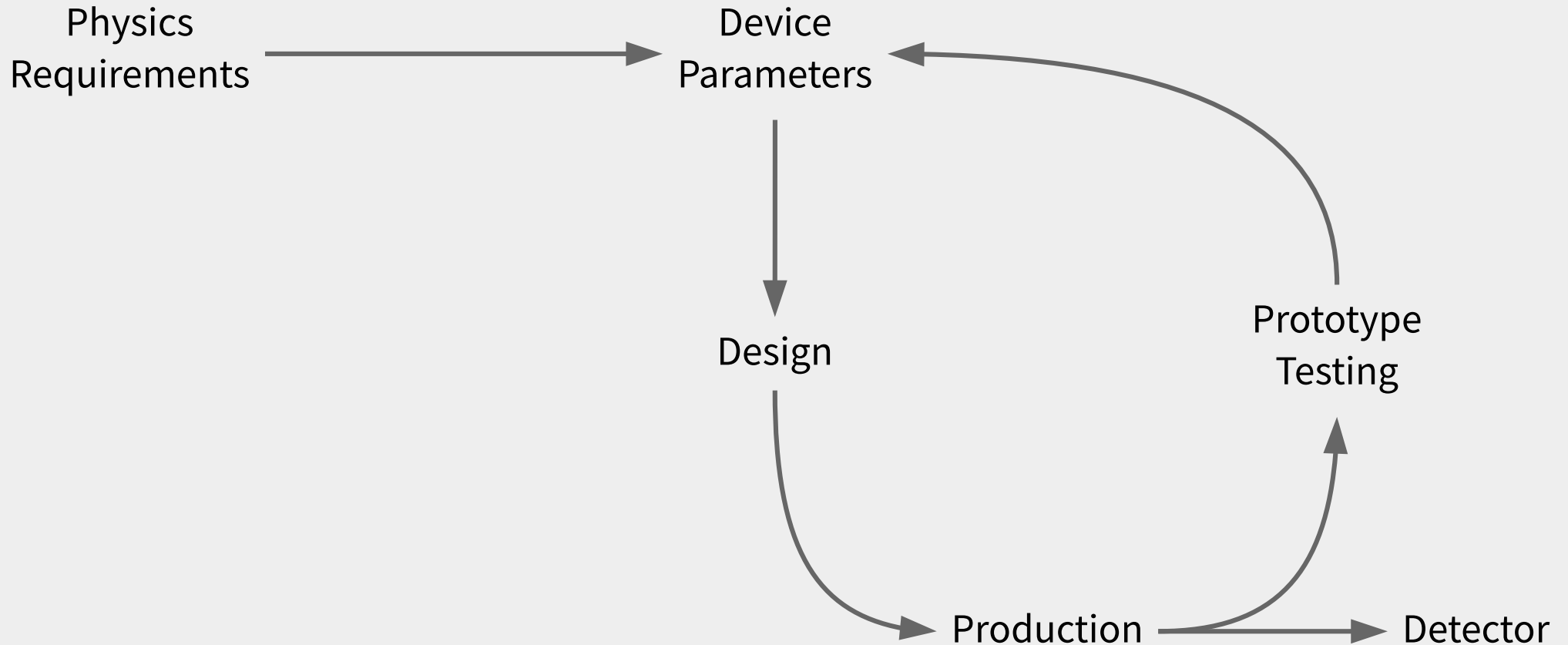
Minimum Ionizing Particle Detector – Broken Down



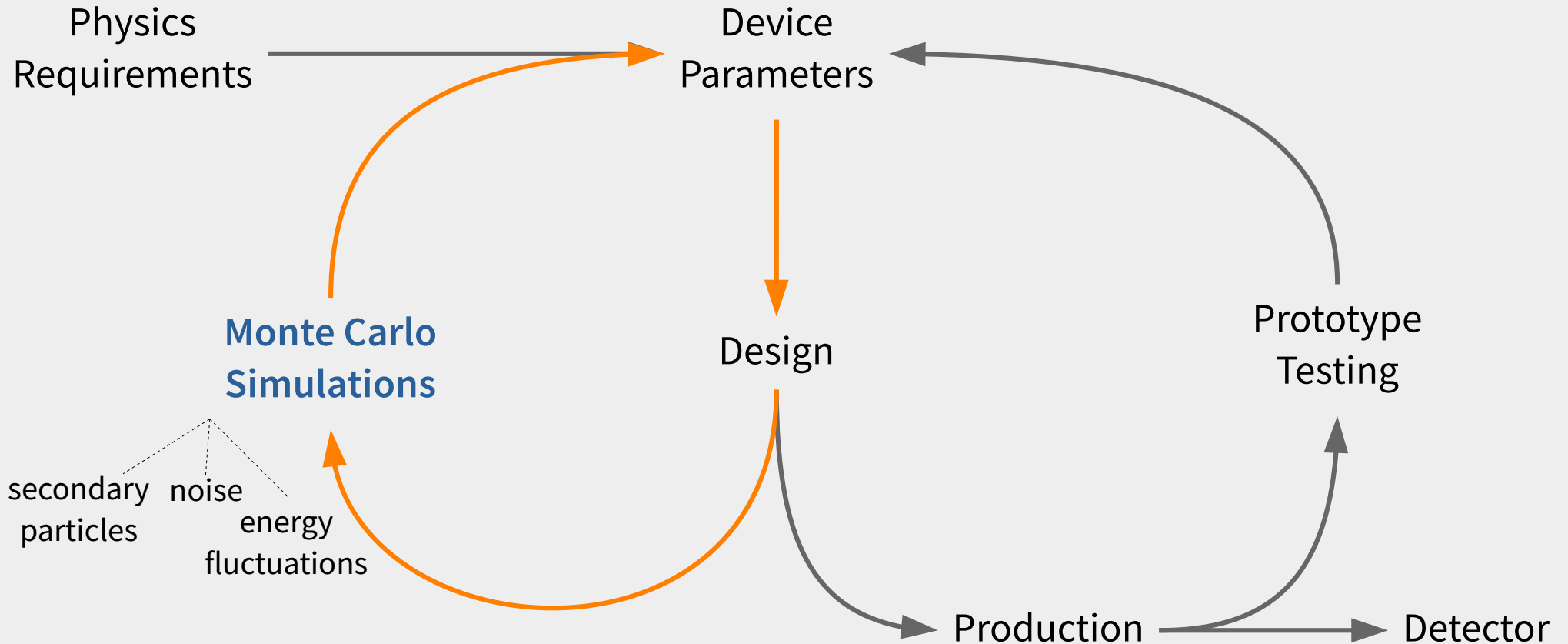
Development Cycle of a Silicon Detector



Development Cycle of a Silicon Detector

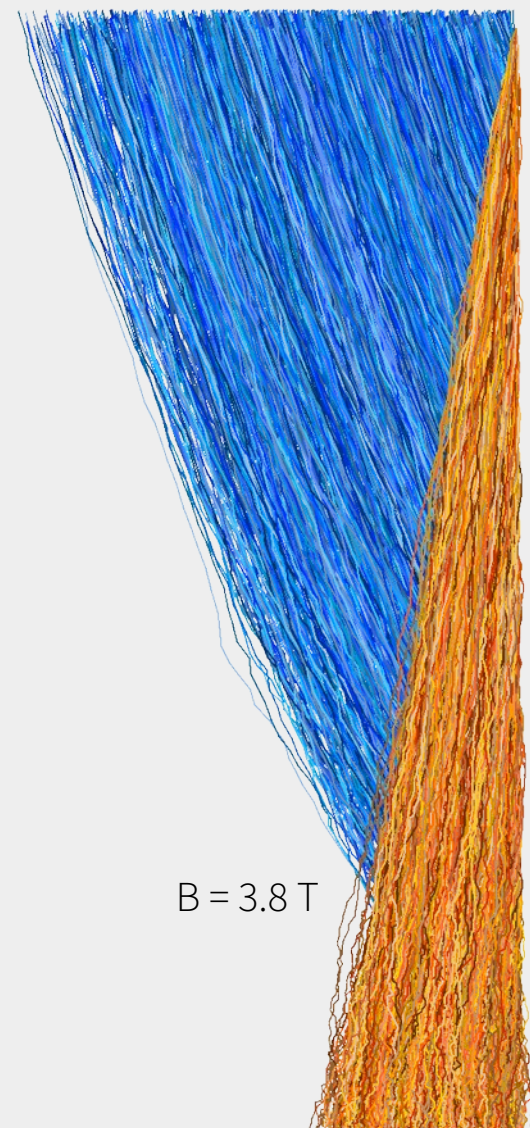


Development Cycle of a Silicon Detector



The Allpix Squared Framework

Monte Carlo Simulation



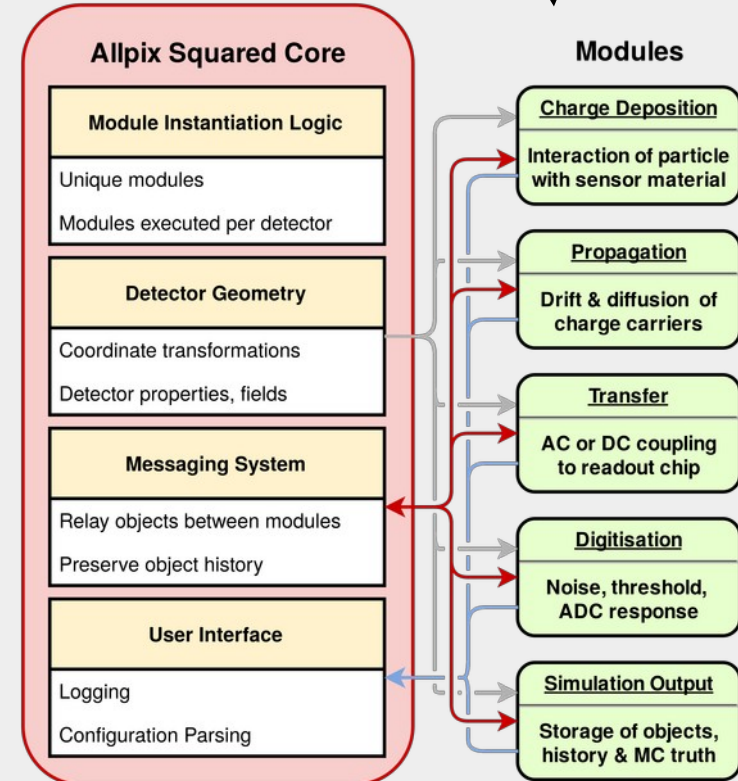
B = 3.8 T

The Allpix² Framework

- **Powerful & flexible**
 - Direct integration with Geant4
 - Many physics models implemented
 - Validated against beam data
- **Easy setup & configuration**
 - Human-readable config files
 - Support for units
 - Fully configurable, no coding required
- Detailed documentation
- **Regular patch & feature releases since 2017**
- **FOSS: MIT-licensed**

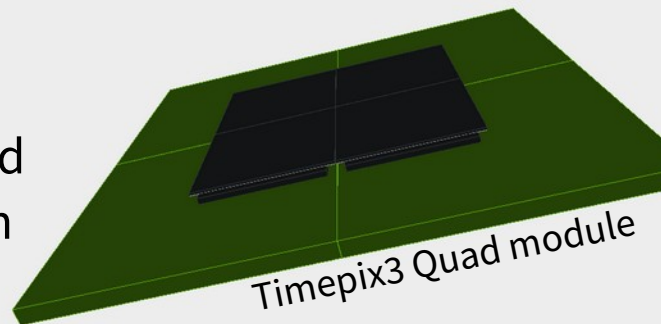
“the rest”

“the physics”



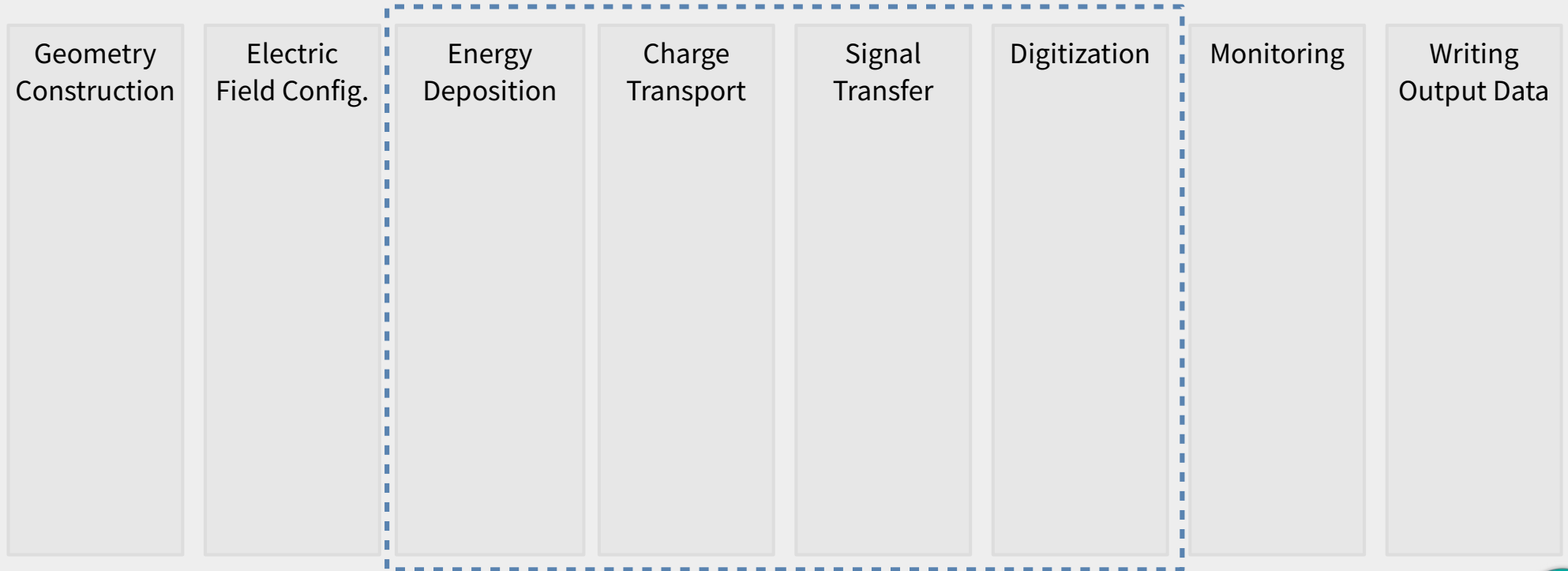
Configuration of the Simulation Chain

- **File 1:** Simulation chain with individual modules
 - Configuration file with modules in order of execution
 - Every parameter documented in manual
- **File 2:** Geometry configuration
 - Position/orientation of individual detectors
 - Model files define detector geometries
 - Geant4 solids generated from model description



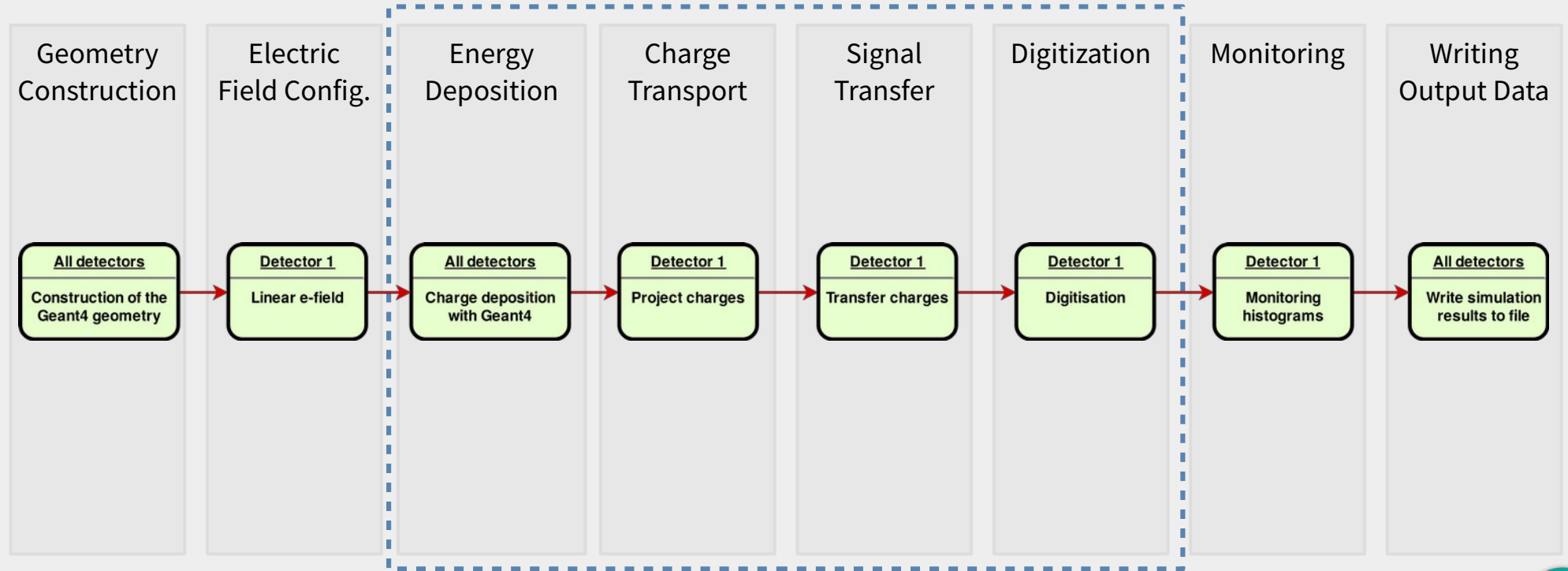
```
1 [AllPix]
2 log_level = "INFO"
3 number_of_events = 500000
4 detectors_file = "telescope.conf"
5
6 [GeometryBuilderGeant4]
7 world_material = "air"
8
9 [DepositionGeant4]
10 physics_list = FTFP_BERT_LIV
11 particle_type = "Pi+"
12 number_of_particles = 1
13 beam_energy = 120GeV
14 # ...
15
16 [ElectricFieldReader]
17 model="linear"
18 bias_voltage=150V
19 depletion_voltage=50V
20
21 [GenericPropagation]
22 temperature = 293K
23 charge_per_step = 10
24 spatial_precision = 0.0025um
25 timestep_max = 0.5ns
26
27 [SimpleTransfer]
```

The Simulation Chain



The Simulation Chain

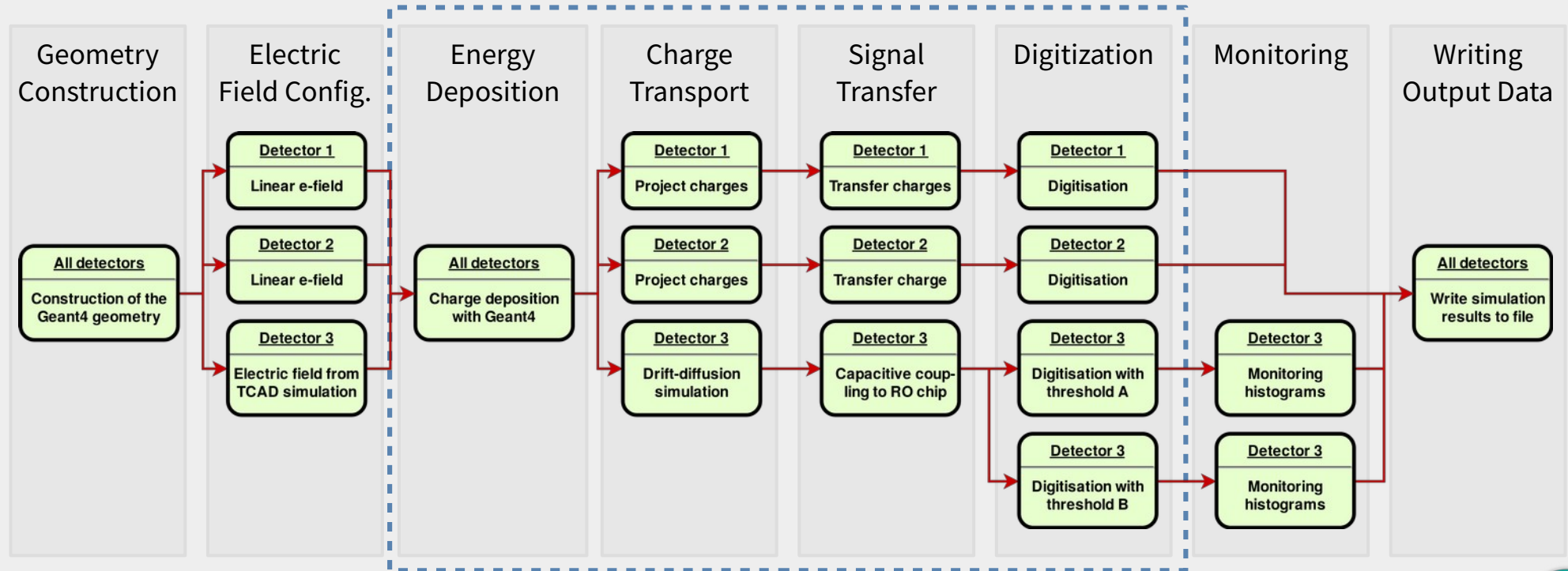
Building blocks follow individual steps of signal formation in detector
 Algorithms for each step can be chosen independently



The Simulation Chain

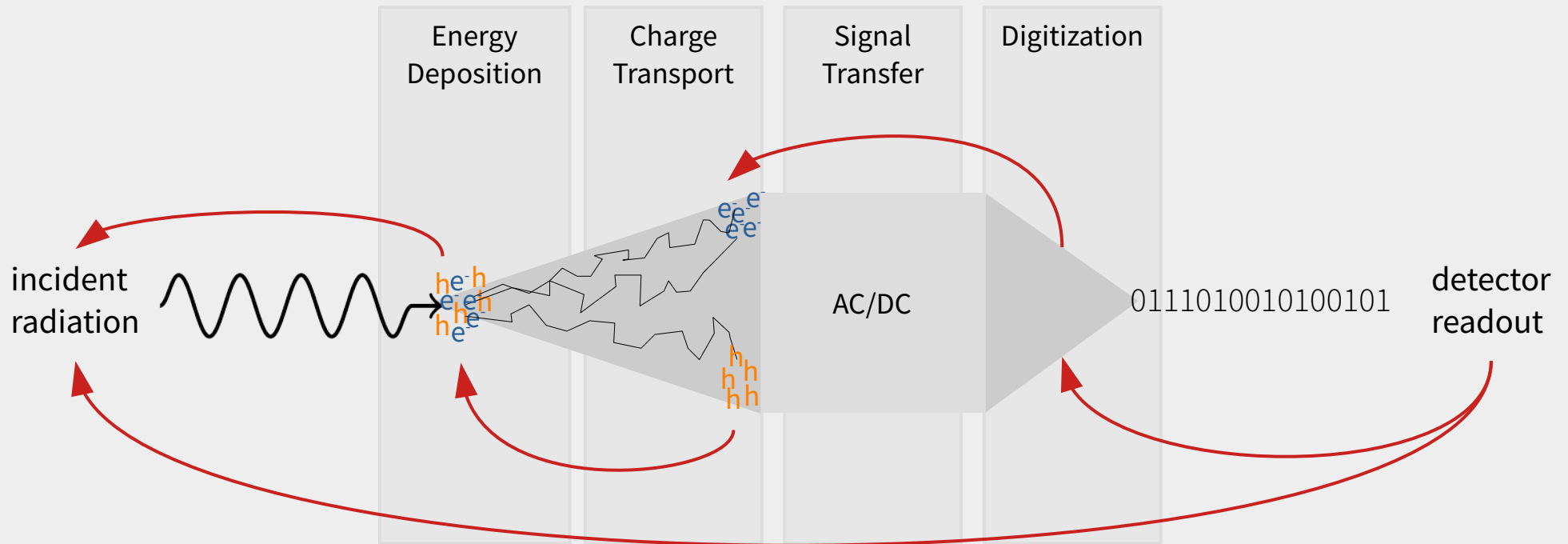
Simulation very flexible: modules configurable on per-detector level

Multiple instances can be run in parallel (simulate different signal formation or front-ends)



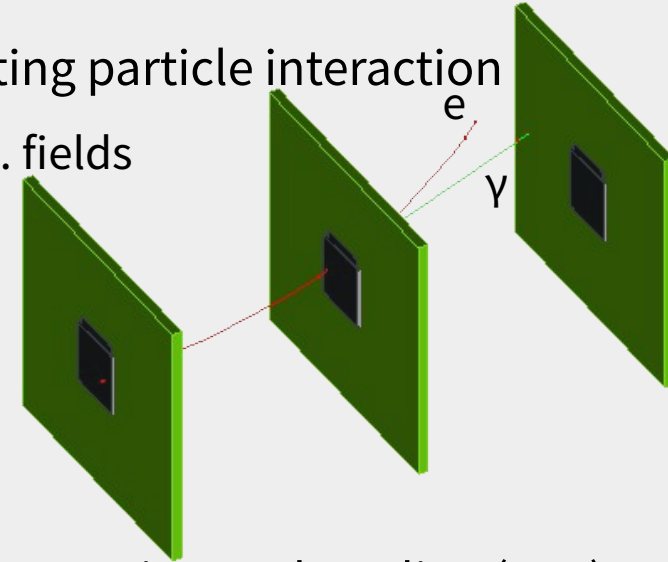
The Monte Carlo Truth

Allpix² keeps history for all simulated objects – available for detailed analysis:



Modules for Energy Deposition

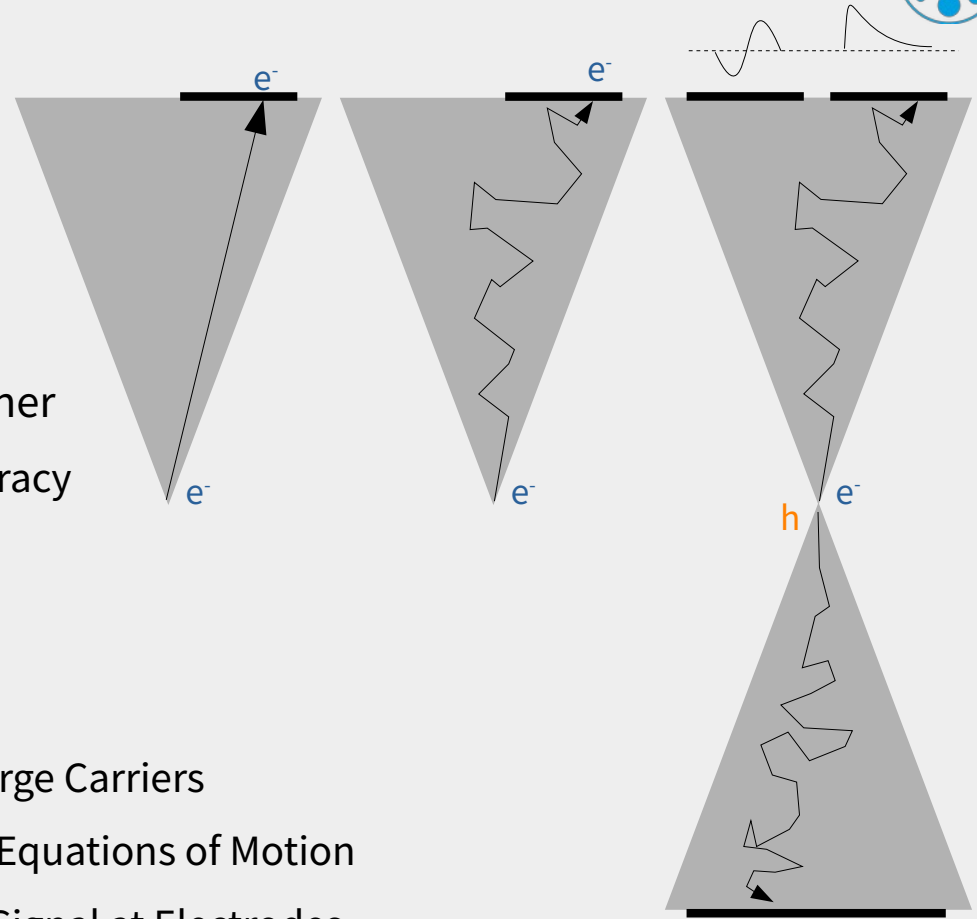
- **DepositionGeant4:** Using established software for simulating particle interaction
 - Tracking of particles through entire setup, including magn. fields
 - Production and tracking of secondary particles
 - Provides MC truth information on all particles
 - Allows visualization of setup
- **DepositionPointCharge:** Simple model, depositing charge at point or along line (LET)
 - Convenient for comparison with e.g. TCAD device simulations
- **DepositionReader:** Read in simulation results from external tools in different formats



Modules for Charge Transport

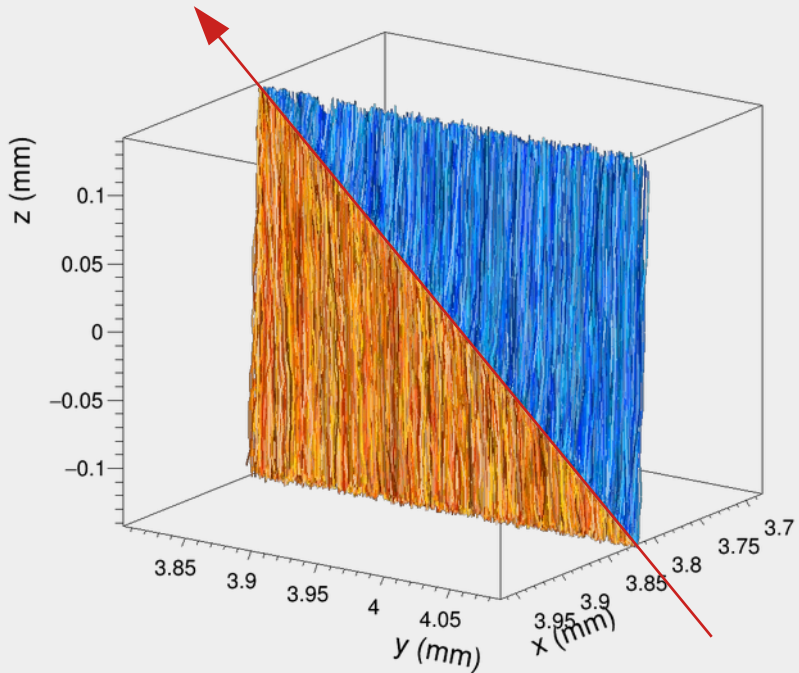


- Most crucial (and time consuming) component in simulation chain
- Multiple charge carriers can be propagated together
 - Depending on initial statistics and required accuracy
 - Some models allow to ignore electrons or holes
- Models with different complexity:
 - **ProjectionPropagation** – $O(1)$, Projecting Charge Carriers
 - **GenericPropagation** – $O(N)$, Integration of Equations of Motion
 - **TransientPropagation** – $O(2 \times N \times M)$, Induced Signal at Electrodes



Drift Path Visualizations

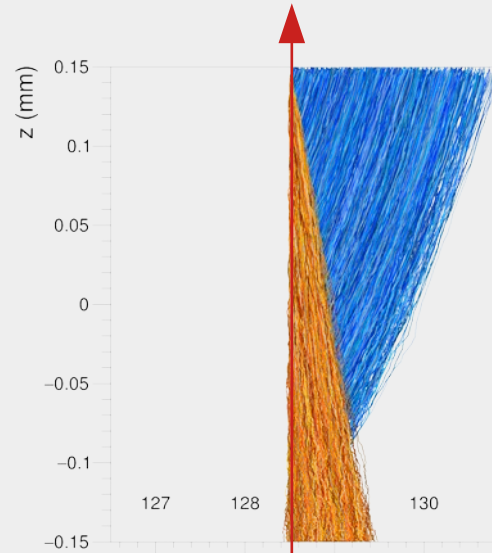
Recording individual steps of the carrier paths enable visualizations



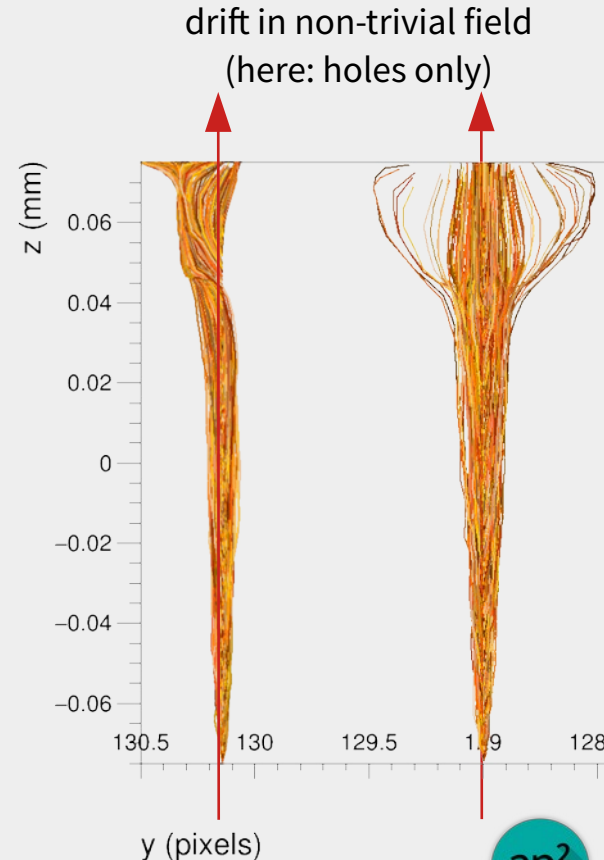
particle at 45° angle
drift paths of electrons & holes



projection along trajectory



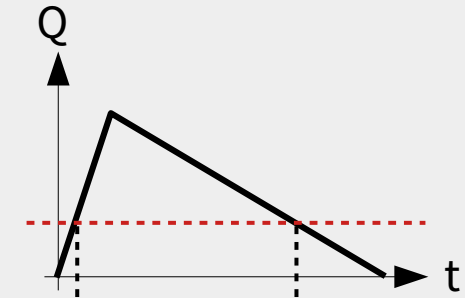
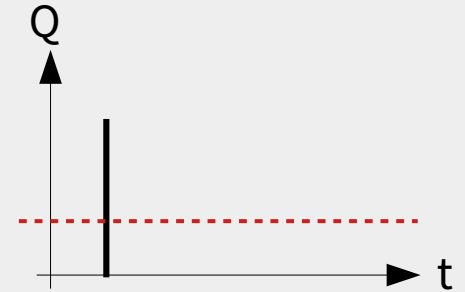
drift in magnetic field:
effect from Lorentz force



drift in non-trivial field
(here: holes only)

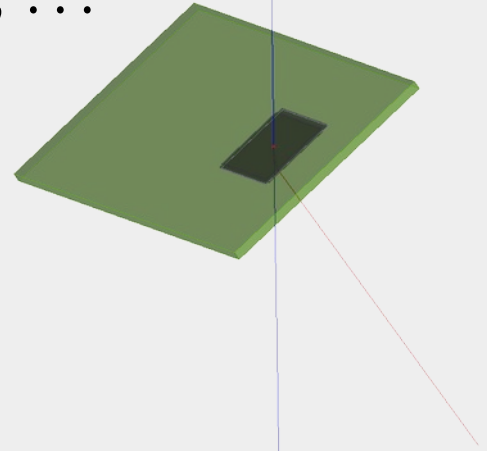
Modules for Digitization

- Methods depend on available information from charge transport
- **DefaultDigitizer:** Simple front-end
 - Compare total charge against configured threshold
 - Add input noise, threshold dispersion, convert to ADC units
 - Possibility to simulate saturation
- **CSADigitizer:** Front-end with timing capabilities
 - Requires current pulse
 - Threshold crossings for time-of-arrival and time-over-threshold
 - Possibility to define custom transfer functions



Application Examples

Detector Systems, CMOS Imaging Sensors, ...



Monolithic CMOS in High-Resistivity Silicon

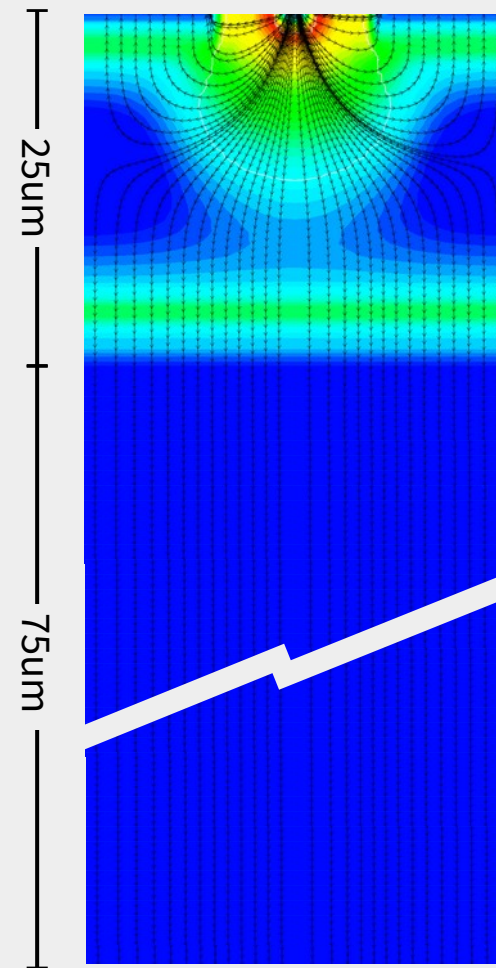
Monolithic sensor in commercial CMOS imaging process

- Pixel pitch 28x28um
- Field in top 25um (high-resistivity) silicon
- Undepleted in 75um silicon substrate

Simulation compared to data taken with 120 GeV π beam

- Simulating only detector under investigation
- Using Monte Carlo truth information as reference

Electrostatic field obtained from TCAD device simulations



Monolithic CMOS in High-Resistivity Silicon

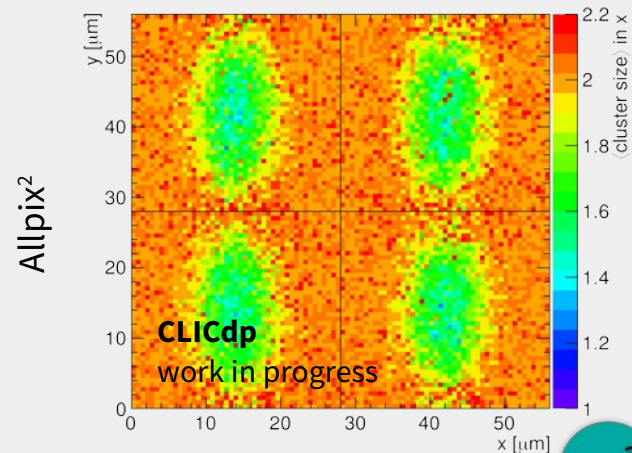
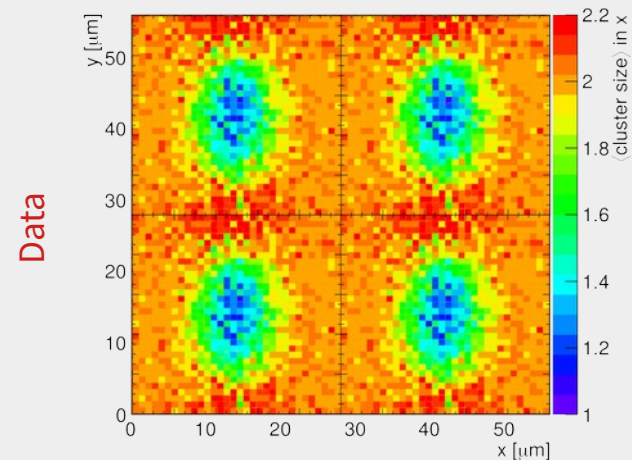
High statistics of 3D Monte Carlo simulation:

- Sampling of quantities within pixel cells
- Here: cluster size in x

Fully depleted planar sensors:
expecting bands without y-dependence

Cluster size exhibits correlation between x/y

- Reason is field configuration & signal contributions from diffusion
- Simulation with TCAD field reproduces correlation

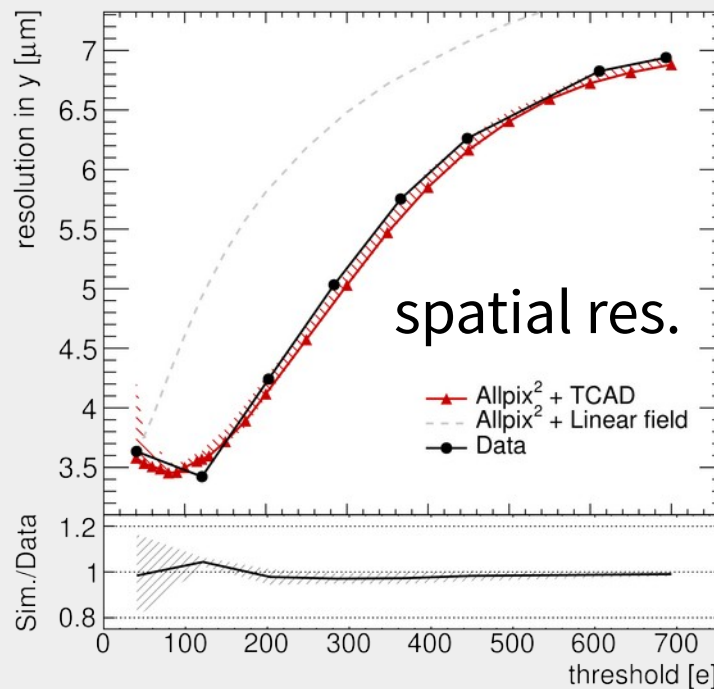
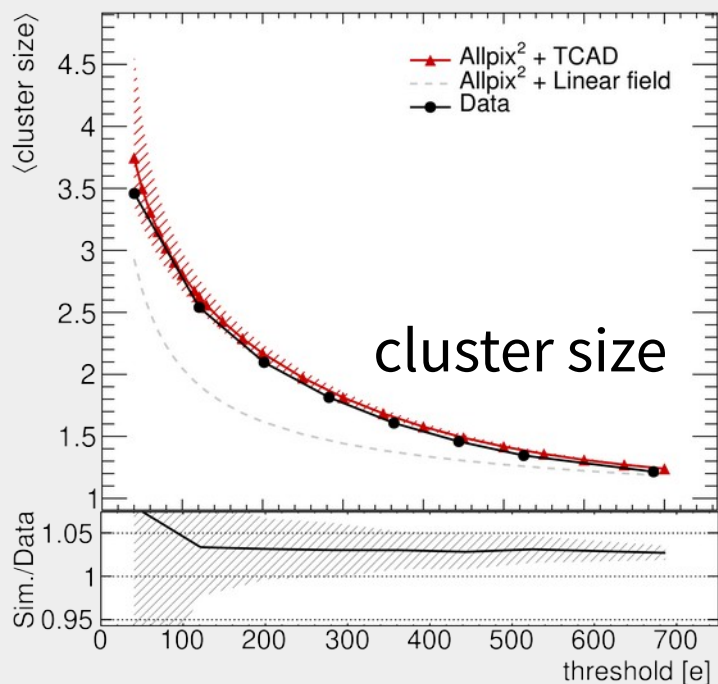




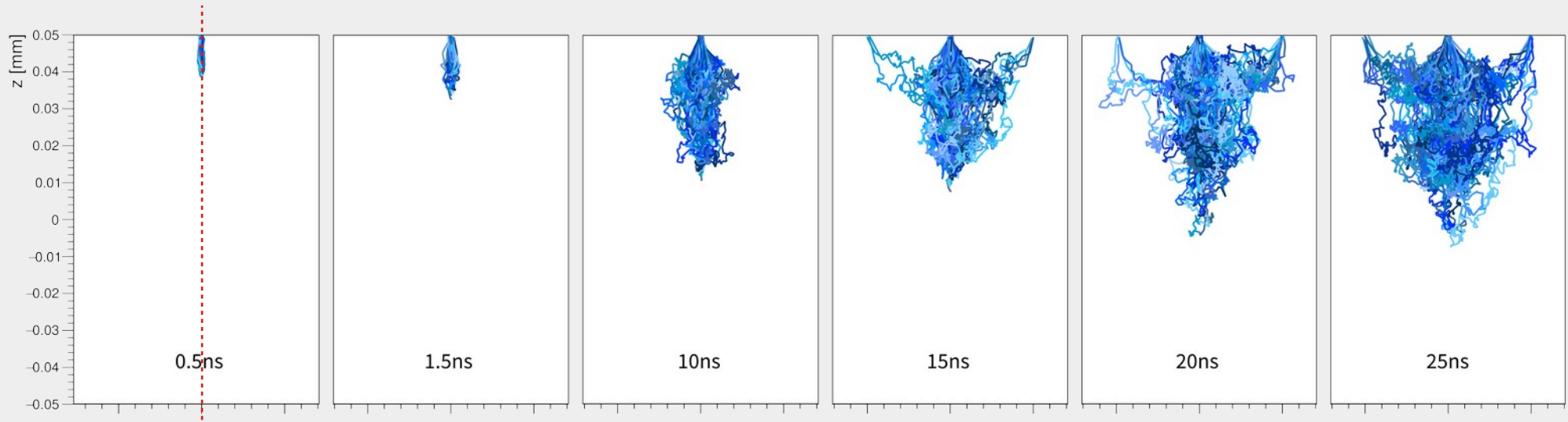
Threshold Dependence of Resolution

Data and simulation match well, e.g. for **cluster size & resolution vs. threshold**

Simulation with linear electric field does not describe data



Visualizing Charge Carrier Motion

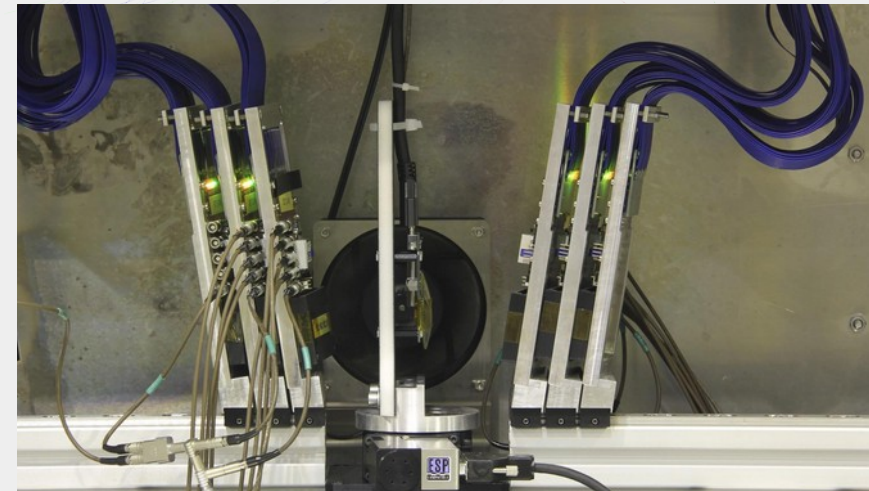
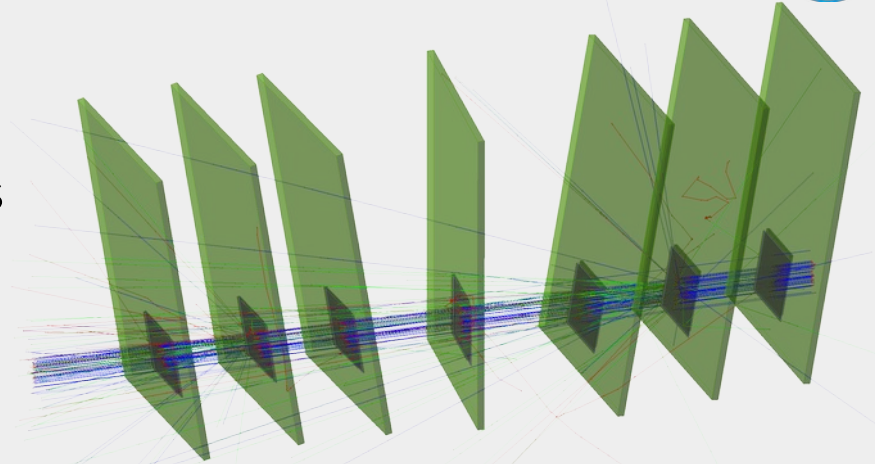


Charge carrier movements at different times after the deposition

- Only electrons shown which reach the electrodes, holes & other electrons omitted
- Contributions from the substrate silicon after ~ 10 ns
- Charge sharing visible after ~ 15 ns

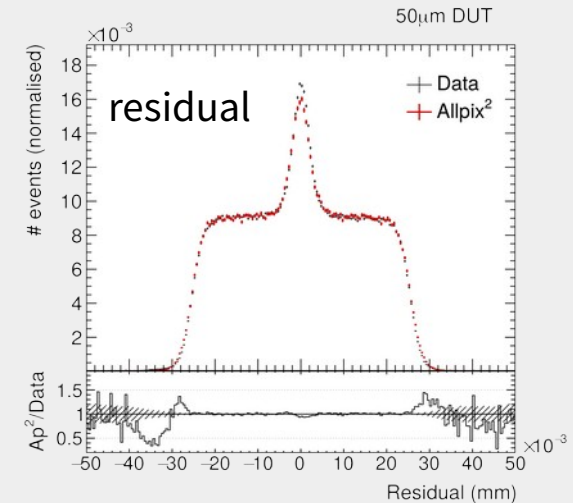
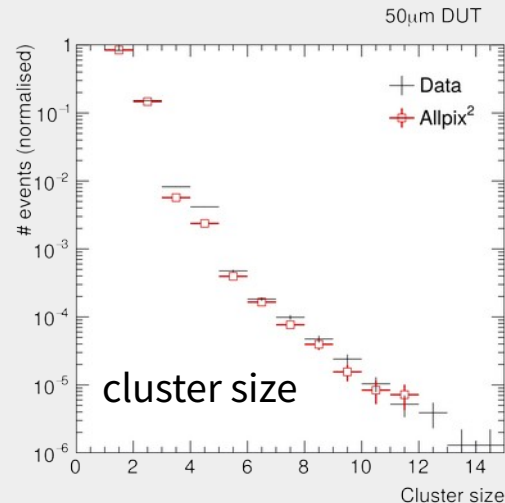
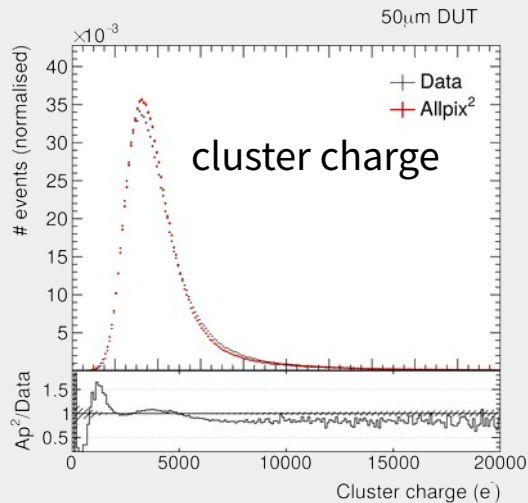
Simulation of a Detector System

- Simulation of a beam telescope setup:
 - Telescope: 6x Timepix3 w/ 300 μm sensors
 - DUT: 1x Timepix3 w/ 50 μm sensor
- Different algorithms used:
 - Telescope: projection
 - DUT: successive integration
- Linear electric field approximation



Simulation of a Detector System

- Using same reconstruction algorithms as for data: clustering, η correction, tracking
- Very good agreement between data and simulation observed (total charge: **Geant4**; cluster size: both; residual shape: **Allpix²**)



Allpix Squared 2.0

Recent Developments

```
Module {
    ModuleManager;
    Messenger;

    /// Brief Base constructor for unique modules
    /// @param config Configuration for this module
    Module(Configuration& config);

    /// Brief Base constructor for detector modules
    /// @param config Configuration for this module
    /// @param detector Detector bound to this module
    /// @note Detector modules should not forget to forward their detector to the base class
    /// @ref InvalidModuleStateException will be raised if the module failed to set up the detector
    Module(Configuration& config, std::shared_ptr<Detector> detector);

    /// Brief virtual destructor.
    virtual ~Module();

    /// @note This method has all delegates linked to this module
    void LinkDelegates();


    /// @note This method is not allowed to be overridden
    virtual void LinkDelegates() = delete;

    /// @note This method is not allowed to be overridden
    virtual void LinkDelegates(const Module&) = delete;

    /// @note This method is not allowed to be overridden
    virtual void LinkDelegates(const Module&) noexcept = delete;
};
```



Release of Allpix Squared 2.0

- First major release introducing structural changes to framework since 1.0 (2018)
 - More than 1500 commits over previous feature release 1.6
 - Introduced fully parallel event processing (Started as  Google Summer of Code project)
- Further separation between physics models & algorithms, Model for mobility, recombination configurable per module instance
 - Mobility: Jacoboni/Canali, Hamburg, Masetti, Arora, ext. Canali/Masetti
 - Recombination: Auger, Shockley-Read-Hall, combined
- Tons of small improvements, cleanup, documentation improvements: <https://cern.ch/allpix-squared/post/2021-06-15-version-2.0.0/>



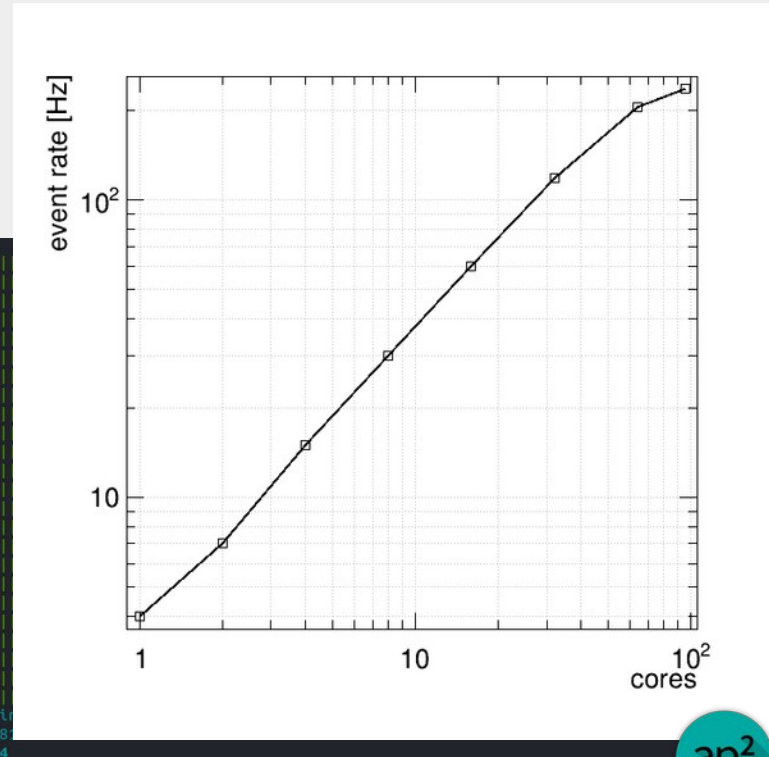
Event-Based Seeding & Multithreading

- Efficient use of system resources / multiple cores
- Retaining strong reproducibility: exact same result, independent of # workers
Event order in output files is preserved
- Fully transparent to user / simulation

```

1 [|||||100.0%] 25 [|||||100.0%] 49 [|||||
2 [|||||99.4%] 26 [|||||98.7%] 50 [|||||
3 [|||||96.8%] 27 [|||||97.4%] 51 [|||||
4 [|||||99.4%] 28 [|||||98.1%] 52 [|||||
5 [|||||98.7%] 29 [|||||99.4%] 53 [|||||
6 [|||||96.8%] 30 [|||||98.7%] 54 [|||||
7 [|||||99.4%] 31 [|||||99.4%] 55 [|||||
8 [|||||100.0%] 32 [|||||97.4%] 56 [|||||
9 [|||||98.1%] 33 [|||||97.4%] 57 [|||||
10 [|||||97.4%] 34 [|||||98.1%] 58 [|||||
11 [|||||95.5%] 35 [|||||98.1%] 59 [|||||
12 [|||||96.8%] 36 [|||||98.7%] 60 [|||||
13 [|||||98.1%] 37 [|||||98.7%] 61 [|||||
14 [|||||97.4%] 38 [|||||98.1%] 62 [|||||
15 [|||||99.4%] 39 [|||||98.7%] 63 [|||||
16 [|||||96.8%] 40 [|||||98.7%] 64 [|||||
17 [|||||97.5%] 41 [|||||99.4%] 65 [|||||
18 [|||||96.8%] 42 [|||||98.7%] 66 [|||||
19 [|||||97.4%] 43 [|||||97.4%] 67 [|||||
20 [|||||99.4%] 44 [|||||99.4%] 68 [|||||
21 [|||||99.4%] 45 [|||||92.9%] 69 [|||||
22 [|||||98.1%] 46 [|||||98.1%] 70 [|||||
23 [|||||98.1%] 47 [|||||98.1%] 71 [|||||
24 [|||||92.9%] 48 [|||||98.1%] 72 [|||||
Mem[|||||16.6G/251G] Tasks: 80, 1073 thr; 96 runni
Swp[|||||846M/4.00G] Load average: 32.64 10.39 3.8
                               Uptime: 180 days(1), 22:26:44
  
```

Allpix² fully utilizing 96 cores



User Manual & Code Documentation

- Focus from the very beginning on well-documented framework
- Source code documentation for every class, method
 - Doxygen markup for code reference
 - Deployed to the website for tags
- Extensive User Manual, describing
 - Framework functionality
 - Physics models
 - Modules & parameters

```
namespace allpix {  
  
    /**  
     * @brief Instantiation of a detector mode  
     *  
     * Contains the detector in the world with  
     * (like the electric field). All model sp  
     * properties are stored in its DetectorMo  
     */  
    class Detector {  
        friend class GeometryManager;  
  
    public:  
        /**  
         * @brief Constructs a detector in the  
         * @param name Unique name of the dete  
         * @param model Model of the detector  
         * @param position Position in the wor  
         * @param orientation Rotation matrix  
         */  
        Detector(std::string name,  
                 std::shared_ptr<DetectorModel>  
                 ROOT::Math::XYZPoint position  
                 const ROOT::Math::Rotation3D&  
  
        /**  
         * @brief Get name of the detector  
         * @return Detector name  
         */  
        std::string getName() const;
```

GenericPropagation

Maintainer: Koen Vanders (k.vanders@cern.ch), Simon Spannagel (s.spannagel@cern.ch)

Status: Functional
Input: EvolvedCharge
Output: PropagatedCharge

Description

Simulates the propagation of electrons and holes through the sensitive volume of the detector. It allows to propagate sets of charge carriers together in order to speed up the simulation while maintaining the required accuracy. The propagation process for these sets is fully independent and no interactions are simulated. The maximum size of the set of propagated charges and thus the accuracy of the propagation can be controlled.

The propagation consists of a combination of drift and diffusion simulation. The drift is calculated using the charge carrier velocity derived from the charge carrier mobility parameter mobility.C. Acceleration at electrodes. The carrier mobility for other electrons or holes is automatically chosen based on the type of the charge carrier under consideration. Thus, also pairs with both electrons and holes is treated properly.

The two parameters `propagate_electrons` and `propagate_holes` allow to control which type of charge carrier is propagated to the respective electrodes. Other one of the carrier types can be selected, or both can be propagated. It should be noted that this will slow down the simulation considerably, since twice as many carriers have to be handled and it should only be used where sensible. The direction of the propagation depends on the electric field configuration, and it should be ensured that the carrier types selected are actually transported to the relevant side for those electric fields, a warning is issued if a possible recombination is detected.

A fourth-order Runge-Kutta-Fehlberg method with fifth order error estimation is used to integrate the electric field. After every Runge-Kutta step, the diffusion is accounted for by applying an offset from a Gaussian distribution calculated from the Drift velocity:

$$r = -\sqrt{2Dt}$$

using the carrier mobility μ , the temperature T and the time step. The propagation stops when the set of charges reaches any surface of the sensor.

The propagation module also produces a variety of output plots. These include a 3D trajectory of the path of all separate propagated charge carrier sets from their point of deposition to the end of their drift, with nearby paths having different colors. In the coloring scheme, electrons are marked in blue colors, while holes are presented in different shades of orange. In addition, a 2D GC is generated for the drift of all individual sets of charges, with the color of the plot corresponding to the number of charges in the set. In addition, the module produces 2D contour simulations in all planes normal to the X, Y and Z axis, showing the concentration flow in the sensor. It should be noted that generating the animations is very time-consuming and should be avoided if even when using a GPU hardware.

Dependencies

This module requires an installation of Diges.

Parameters

- `temperature`: Temperature of the sensitive device, used to estimate the diffusion constant and therefore the strength of the diffusion. Defaults to room temperature (300.15K).
- `charge_per_step`: Maximum number of charge carriers to propagate together. Overrides the total number of deposited charge carriers at a specific point in time and the number of charge carriers used, with the remaining charge carriers. A value of 10 charges per steps usually default. If this value is not specified.
- `spatial_precision`: Spatial precision to use for the re-estimation of the Runge-Kutta propagation to adjust to re-estimate spatial precision after calculating the necessary from the fifth order error method. Defaults to 0.1 cm.
- `time_step_start`: Time step to initiate the Runge-Kutta integration with. A progressive estimation of the parameter reduces the time to optimize the time step to the spatial precision parameter. Default value 0.01 ns.
- `time_step_min`: Minimum step size to use for the Runge-Kutta integration regardless of the spatial precision. Defaults to 1 ps.
- `time_step_max`: Maximum step size to use for the Runge-Kutta integration regardless of the spatial precision. Defaults to 0.1 ns.
- `integration_order`: Time with which charge carriers are propagated. After exceeding this time, no further propagation is performed for the respective carriers. Defaults to the LHC bunch crossing time of 25ns.
- `propagate_electrons`: Select whether electron-type charge carriers should be propagated to the electrodes. Defaults to false.
- `propagate_holes`: Select whether hole-type charge carriers should be propagated to the electrodes. Defaults to false.
- `output_plots`: Determine if output plots should be generated for every event. This causes a significant slow-down of the simulation, it is not recommended to enable this option for runs with more than a couple of events. Default is false.
- `output_plots_time`: Time step to calculate the necessary for points plotted. Indirectly determines the amount of points plotted. Defaults to inverse of `charge_per_step` if not explicitly specified.
- `output_plots_theta`: Viewpoint angle of the 3D animation and the 2D line graph around the world X-axis. Defaults to zero.
- `output_plots_phi`: Viewpoint angle of the 3D animation and the 2D line graph around the world Z-axis. Defaults to zero.
- `output_plots_use_equal_scaling`: Determine if the plots should use pixels as unit instead of metric length scales. Defaults to false (i.e. using the metric system).
- `output_plots_use_equal_scaling`: Determine if the plots should be produced with equal distance scales on every axis (i.e. if this is true it implies that some points will fall out of the graph). Defaults to true.
- `output_plots_align_pixels`: Determine if the plot should be aligned on pixels. Defaults to false if enabled the start and the end of the axis will be at the edge pixel between pixels.
- `output_animations`: In addition to the other output plots, also write a GIF animation of the charge drifting towards the electrodes. This is very slow and writing the animation takes a considerable amount of time, therefore defaults to false. This option also requires `output_plots` to be enabled.
- `output_animations_line_scaling`: Scaling for the animation used to convert the actual animation time to the time step of the animation. Defaults to 1, but meaning that every frame of the animation requires an animation of a single second.
- `output_animations_color_max`: Scaling for the markers on the animation, defaults to one. The markers are already internally scaled to the charge of the step, normalized to the maximum charge.
- `output_animations_color_max_scaling`: Scaling to use for the color scale axis from the theoretical maximum charge at every single plot step. Default is 10, meaning that the maximum of the color scale axis is equal to the total amount of charges (i.e. `charge_per_step` multiplied by the number of steps). Parameter can be used to improve the color scale of the contour plots.
- `output_animations_color_markers`: Determine if colors should be for the markers in the animations, defaults to false.

Usage

An example of generic propagation for all sensors of type "Trapez" at room temperature using pixels of 25 charges in the following:

```
{GenericPropagation}  
type = "Trapez"  
temperature = 300K  
charge_per_step = 25
```

2nd Allpix Squared User Workshop

Workshop: 17 – 19 August 2021
<https://indi.to/WhdJn>

Abstract submission: 24 July 2021

Registration: 16 August 2021

Contributions welcome!



2nd Allpix² User Workshop

17-19 August 2021 Wherever you are (virtual)

Deadlines: Abstract submission: 24 July 2021
Registration: 16 August 2021

Organizers:
Adriana Simancas (DESY)
Anastasiia Velyka (DESY)
Katharina Dort (CERN)
Paul Schütze (DESY)
Simon Spannagel (DESY)

- **New Features**
- **Software Development**
- **Expert Talks**
- **User Applications & Studies**
- **Simulation Case Studies**

For registration and abstract submission, please scan the QR code or go to: <https://indi.to/WhdJn>



Summary

- Monte Carlo simulations:
vital component of understanding & interpreting detector performance
- Allpix²: comprehensive MC simulation framework for silicon detectors
 - Offers a variety of modules which can be combined to form simulation chain
 - Easy to use and well documented
 - Validation against data recorded in particle beams
- Continuous development and support
 - New major release last week, providing full multithreading capabilities
 - Regular patch releases with bug fixes
- 2nd Allpix Squared User workshop 17 – 19 August 2021

Allpix Squared Resources



Website

<https://cern.ch/allpix-squared>



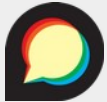
Repository

<https://gitlab.cern.ch/allpix-squared/allpix-squared>



Docker Images

https://gitlab.cern.ch/allpix-squared/allpix-squared/container_registry



User Forum:

<https://cern.ch/allpix-squared-forum/>



Mailing Lists:

allpix-squared-users <https://e-groups.cern.ch/e-groups/Egroup.do?egroupId=10262858>

allpix-squared-developers <https://e-groups.cern.ch/e-groups/Egroup.do?egroupId=10273730>



User Manual:

<https://cern.ch/allpix-squared/usermanual/allpix-manual.pdf>

