

PAPER • OPEN ACCESS

CaloClouds: fast geometry-independent highly-granular calorimeter simulation

To cite this article: Erik Buhmann *et al* 2023 *JINST* **18** P11025

View the [article online](#) for updates and enhancements.

You may also like

- [LiDAR-SLAM loop closure detection based on multi-scale point cloud feature transformer](#)
Shaohua Wang, Dekai Zheng and Yicheng Li
- [Enhanced curve-based segmentation method for point clouds of curved and irregular structures](#)
Limei Song, Zongyang Zhang, Chongdi Xu et al.
- [Point cloud simplification algorithm based on the feature of adaptive curvature entropy](#)
Guolin Wang, Lushen Wu, Yun Hu et al.



PRIME
PACIFIC RIM MEETING
ON ELECTROCHEMICAL
AND SOLID STATE SCIENCE

HONOLULU, HI
Oct 6–11, 2024

Abstract submission deadline:
April 12, 2024

Learn more and submit!



Joint Meeting of

The Electrochemical Society
•
The Electrochemical Society of Japan
•
Korea Electrochemical Society

CaloClouds: fast geometry-independent highly-granular calorimeter simulation

Erik Buhmann,^a Sascha Diefenbacher,^{a,b} Engin Eren,^c Frank Gaede,^{c,d}
Gregor Kasieczka,^{a,d} Anatolii Korol,^{c,*} William Korcari,^a Katja Krüger^c
and Peter McKeown^c

^a*Institut für Experimentalphysik, Universität Hamburg,
Luruper Chaussee 149, 22607 Hamburg, Germany*

^b*Physics Division, Lawrence Berkeley National Laboratory,
1 Cyclotron Rd, Berkeley, CA 94720, U.S.A.*

^c*Deutsches Elektronen-Synchrotron DESY,
Notkestr. 85, 22607 Hamburg, Germany*

^d*Center for Data and Computing in Natural Sciences CDCS, Deutsches Elektronen-Synchrotron DESY,
Notkestr. 85, 22607 Hamburg, Germany*

E-mail: anatolii.korol@desy.de

ABSTRACT: Simulating showers of particles in highly-granular detectors is a key frontier in the application of machine learning to particle physics. Achieving high accuracy and speed with generative machine learning models would enable them to augment traditional simulations and alleviate a major computing constraint. This work achieves a major breakthrough in this task by, for the first time, directly generating a point cloud of a few thousand space points with energy depositions in the detector in 3D space without relying on a fixed-grid structure. This is made possible by two key innovations: i) Using recent improvements in generative modeling we apply a diffusion model to generate photon showers as high-cardinality point clouds. ii) These point clouds of up to 6,000 space points are largely geometry-independent as they are down-sampled from initial even higher-resolution point clouds of up to 40,000 so-called GEANT steps. We showcase the performance of this approach using the specific example of simulating photon showers in the planned electromagnetic calorimeter of the International Large Detector (ILD) and achieve overall good modeling of physically relevant distributions.

KEYWORDS: Simulation methods and programs; Data processing methods; Analysis and statistical methods; Calorimeter methods

ARXIV EPRINT: [2305.04847](https://arxiv.org/abs/2305.04847)

*Corresponding author.



Contents

1	Introduction	1
2	Data samples	2
3	Model	4
3.1	Diffusion point cloud generator	6
3.2	Encoder & latent flow	8
3.3	Shower flow & post-diffusion calibration	9
4	Results	10
4.1	Physics performance	10
4.2	Shower translation	12
4.3	Timing	12
5	Conclusions	14
A	Effects of the pre-clustering	15

1 Introduction

Large-scale experiments in high energy physics (HEP) have one unifying feature: the need to record and analyze an ever-increasing amount of data produced primarily through higher collider luminosities and higher granularity detectors with growing numbers of readout channels. The reduced experimental uncertainty that comes with this increased amount of higher-resolution measurement data presents one of the most promising avenues to increase the precision of measurements and the sensitivity of searches for new physical phenomena beyond the Standard Model.

In order to compare experimental measurement and theoretical prediction, modern HEP experiments rely on high-precision Monte Carlo (MC) simulations. These simulations model collider events from the initial hard scattering, over hadronization, to the detailed responses of the tracking-, calorimeter-, and muon-systems of the detector. Such detailed modeling is a highly resource-intensive process. Further, for subsequent comparison to collision data to be valid, the number of simulated events has to at least equal the number of recorded collisions. The combination of these two factors means that MC simulation alone places a significant strain on the available HEP computing resources, which, for future high-luminosity experiments, is only bound to increase further [1, 2].

The most notable bottleneck within the MC simulation chain is the step that has to deal with a large number of particles interacting with complex detectors — shower simulation — as the modeling of all involved particles is inherently time-consuming [3]. This provides a strong incentive to find new ways of speeding up MC simulations. One of the most promising current fast simulation

approaches is provided by generative machine learning (ML) models. These models can learn an underlying distribution from a given data set and can subsequently be used to generate new data from the learned distribution, allowing them to generate simulated data potentially orders of magnitude faster than classical simulations.

The concept of generative calorimeter simulation is well-proven at this point. Its underlying mathematical viability has been demonstrated [4, 5] and direct applications have been shown using generative adversarial networks [6–15], autoencoder-variants [16–18], normalizing flows [19–21] and diffusion models [22–24]. They have been successfully deployed e.g. by the ATLAS collaboration [25].

Previous generative calorimeter models focused on fixed geometrical structures, either in the form of 3-dimensional tensors that mirror the design of a calorimeter or in the form of an ordered list, that connects one output to one calorimeter cell. Within the fixed-structure framework, applications have started out using comparatively simplistic calorimeter geometries and gradually moved to higher and higher detector granularity. This directly follows the advances in calorimeter development, which similarly aim for increasingly granular detectors [26, 27].

It is, however, this move to higher granularity, that presents the largest challenge for fixed-structure generative models, as higher data dimensionality entails more resource-intensive generation. Most notably, the increased sparsity of highly granular detector data means a large fraction of the model evaluation is *wasted* on empty cells. In an ideal scenario, this wasted computation could be avoided and only the non-empty sections of the calorimeter would be simulated.

To overcome this issue, we introduce CALOCLOUDS, a generative shower simulation approach that does not rely on a fixed structure, but instead generates geometry-independent point clouds.¹ This approach removes the need to simulate empty detector regions. Further, the produced point clouds can be directly projected into an arbitrary detector geometry, thereby allowing for effortless translation invariance and making the handling of e.g. non-regular grids or hexagonal structures straightforward.

Point cloud-based generative models have previously been explored in HEP, most notably for fast hadronization simulation [24, 28, 29], however, this work is the first to successfully expand this technique to handle the significantly larger point clouds of several thousand space points required for realistic calorimeter simulations.

In the following, we describe the data in section 2, introduce the model in section 3, show results in section 4, and offer conclusions in section 5.

2 Data samples

We investigate the performance of our generative point cloud model with a purpose-built dataset that describes the energy depositions of photon showers in the electromagnetic calorimeter (ECAL) of the International Large Detector (ILD) [30]. The calorimeter data training set consists of 524k² photon showers, with incident energy uniformly distributed between 10 GeV and 90 GeV, simulated in the ECAL of the ILD. The ILD ECAL is a highly granular sampling calorimeter, consisting of 30 layers with alternating passive tungsten absorbers and active silicon sensors. The first (as encountered by the shower) 20 absorber layers have a thickness of 2.1 mm, while the last 10 absorbers have a thickness of 4.2 mm. Every silicon layer has a thickness of 0.5 mm and subdivided into 5 mm × 5 mm cells, which

¹The code is available at <https://github.com/FLC-QU-hep/CaloClouds>.

²The training data is available at <https://zenodo.org/records/10044175>.

are read out individually. The showers are simulated using `GEANT4` Version 10.4 (with `QGSP BERT` physics list) in a detailed simulation model of the ILD detector implemented in the `iLCSoft` [31] framework. In the case of the ILD ECAL, the model implemented in `DD4HEP` [32] includes realistic gaps between silicon sensors and staggering between layers, such that the cells form an irregular (position-dependent) grid in 3D space. In order to describe the showers, we introduce two coordinate systems: a compact area near the photon’s impact position within the calorimeter is denoted by the coordinates $[X, Y, Z]$, with X and Y pointing parallel to the orientation of the calorimeter layers and Z pointing perpendicular to the calorimeter layers into the detector along the direction of the shower propagation and the $[X', Y', Z']$ coordinates that constitute the global coordinate system used in ILD, with Z' pointing parallel to the beam pipe, X' laying in the horizontal plane and Y' pointing vertically upwards.

The simulated photons are produced at $[X' = 0, Y' = 1811.3 \text{ mm}, Z' = 40 \text{ mm}]$ with a trajectory along Y' , i.e. they are created right at the front of the calorimeter in order to avoid interactions before entering the calorimeter, as well as being positioned so as to avoid cracks in the detector.

During the full simulation with `GEANT4`, a very large number of individual energy depositions (on average 20,000 per shower) from secondary particles traversing the sensors is created in the sensitive materials and then added up in the actual calorimeter cells of $5 \text{ mm} \times 5 \text{ mm}$ transversal size. While in principle these steps³ could be used as input to the point cloud model, their number is prohibitively large, and therefore a pre-clustering procedure is applied before the actual training. However, this clustering still needs to be more granular than the physical cell size to allow a shower projection into any part of the calorimeter (except changing its depth) without introducing reconstruction artefacts due to cell staggering and gaps.

For this pre-clustering procedure, the steps are grouped by their layer number and then projected into a regular grid with a 36 times higher granularity than that of the actual calorimeter, i.e. into grid cells of $0.83 \text{ mm} \times 0.83 \text{ mm}$ transversal size, thereby reducing the total number of space points by a factor of circa 7. The effects of this clustering are largely negligible when considering complete showers, as is shown in detail in A. Note, that only active steps are simulated, maintaining the main advantage of point cloud-based simulation.

In order to normalize the range of inputs to the model, we define a bounding box around the showers. This box extends from -200 mm to 200 mm in X and Y around the fixed impact point and spans across all layers, making it around seven times more voluminous than what was used in previous ILD ECAL photon shower data sets [16]. Any clusters that still end up outside the bounding box — less than 4 per mille — are discarded. Finally, as there can be no recorded hits in the absorber layers, the data is transformed such that the absorber regions are removed and the active layers become contiguous. Furthermore, the Z positions within the active layers are uniformly smeared within a layer to produce data that is smooth in Z , whereas in the ILD simulation, the energy is deposited for purely technical reasons predominantly at the center of a layer. The cluster positions are then normalized such that the opposing boundaries of the bounding box correspond to a position of -1 and 1 respectively. Figure 1 presents an overview of the data preprocessing pipeline.

In addition to the training data, two validation data sets and one test set were simulated. The two validation data sets were simulated at the same position as the training data, the first containing 40k samples with energies uniformly distributed between 10 GeV and 90 GeV , and the

³Following the standard `GEANT4` [33] terminology.

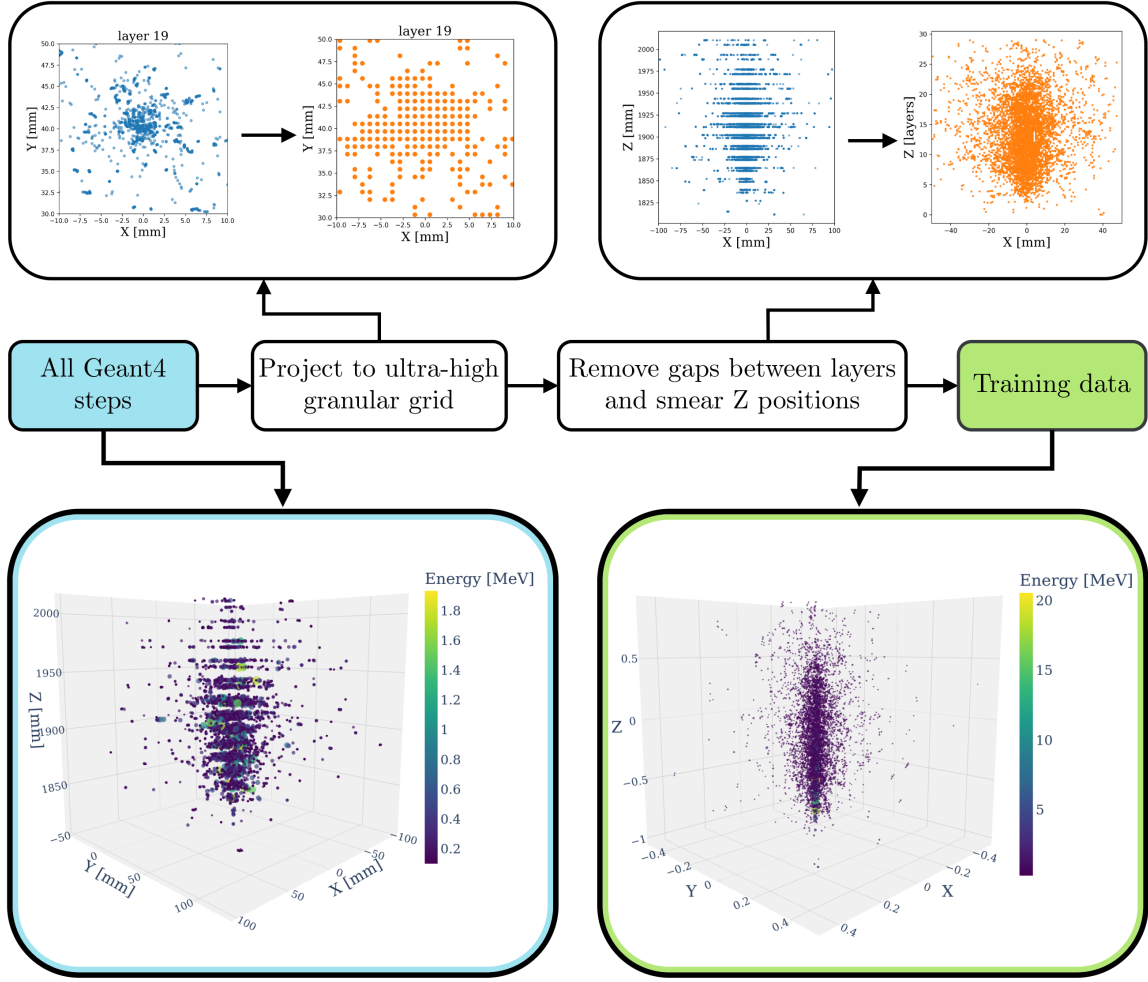


Figure 1. Illustration of the data preprocessing pipeline.

second consisting of fixed energy photon showers with incident particle energies of 10 GeV, 50 GeV, and 90 GeV. Each set with fixed energy contains 2k showers, totaling 6k data samples. Finally, the test set contains 2k samples with fixed incident particle energy of 50 GeV that differs in the position at which the photons were simulated. The validation sets have the photons produced at $[X' = 0, Y' = 1811.3 \text{ mm}, Z' = 40 \text{ mm}]$, matching the training data, while the test set places the origin of the photons at $[X' = 37.5 \text{ mm}, Y' = 1811.3 \text{ mm}, Z' = -36.1 \text{ mm}]$. This allows us to investigate the behavior of the generated point clouds under a translation to another position in the calorimeter, where the local cell geometry is different (see section 4.2).

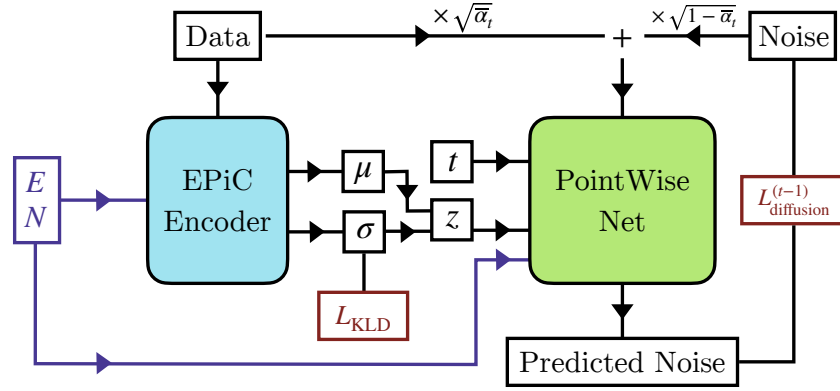
Finally, table 1 provides an overview of the types of point clouds considered in this work.

3 Model

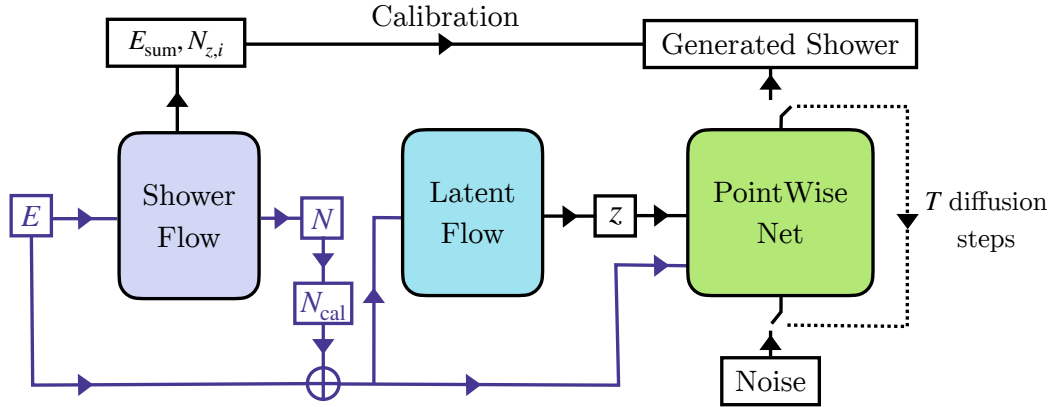
Point cloud generative models take a given number of random noise points and transform them into a point cloud of the desired shape (potentially with additional point features as in our case). This makes producing clouds with a variable number of points inherently challenging. To address this

Table 1. Overview of the three different types of point clouds considered in this work, indicating their role. Note that the number of points per shower (second column) only provides an order of magnitude.

	points / shower	Note
All GEANT4 steps	40 000	Initial output of GEANT4
Clustered GEANT4 steps	6 000	Input/output of CALOCLOUDS
Hits in calorimeter grid	1 000	Calculation of physics observables



(a) Training at random time step t .



(b) Sampling with reverse diffusion through all time steps T .

Figure 2. Illustration of the training and sampling procedure of the CALOCLOUDS architecture. The separate training of the Shower Flow and the Latent Flow is not shown.

cardinality issue, we employ a two-step approach during model training and a three-step approach during sampling, drawing inspiration from ref. [34]. In the following, we first briefly outline the overall structure with additional details provided in corresponding subsections. An illustration of the training and sampling procedure is shown in figure 2.

During training, the original data is first encoded with a Variational-Autencoder (VAE)-like *EPiC Encoder*, conditioned on the incident particle energy and the number of points, into a near-Gaussian

distributed latent space. Second, this latent space, together with the incident energy and number of points, is used in a conditional point cloud diffusion model termed *PointWise Net*.

During sampling, the encoded latent space is generated with a conditional *Latent Flow* model. Since this Latent Flow needs to be conditioned on the incident energy and the number of points, a second *Shower Flow* is employed during sampling to generate an appropriate number of points from a requested incident energy. This way, the only conditional variable for the whole model is the particle incident energy E . Additionally, the Shower Flow generates the total visible energy of the calorimeter point cloud E_{sum} as well as the number of points per layer $N_{z,i}$ for a post-diffusion calibration of the generated point cloud.

In section 3.1 we discuss first the PointWise Net at the core of the overall proposed CALOCLOUDS architecture. Then, in section 3.2, we introduce the EPiC Encoder used during training and the Latent Flow used to perform sampling. Finally, in section 3.3 we outline the Shower Flow used to generate the conditioning variables as well as calibration factors.

3.1 Diffusion point cloud generator

The main part of the CALOCLOUDS architecture is the PointWise Net which is trained as a diffusion probabilistic model and was introduced in ref. [34]. In general, diffusion models [35] learn a Markov chain of Gaussian distortions that over T time steps diffuse data to noise. Here the data is a point cloud $X^{(0)} = \{\mathbf{x}_i^{(0)}\}_{i=1}^N$ comprised of N points with the exponent in parentheses denoting the time step.

During training, a forward diffusion process for every point is modeled as a Markov chain:

$$\begin{aligned} q(\mathbf{x}_i^{1:T} | \mathbf{x}_i^{(0)}) &= \prod_{t=1}^T q(\mathbf{x}_i^{(t)} | \mathbf{x}_i^{(t-1)}), \\ q(\mathbf{x}_i^{(t)} | \mathbf{x}_i^{(t-1)}) &= \mathcal{N}(\mathbf{x}_i^{(t)} | \sqrt{1 - \beta_t} \mathbf{x}_i^{(t-1)}, \beta_t \mathbf{I}), \end{aligned} \quad (3.1)$$

for time steps $t \in 1, \dots, T$, \mathcal{N} denoting the normal distribution, and variance schedule $\beta_1 \cdots \beta_T$ controlling the diffusion process. To generate realistic point clouds from a simple noise distribution, the reverse diffusion process starting from noise $p(\mathbf{x}_i^{(T)}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ is defined as:

$$\begin{aligned} p_\theta(\mathbf{x}_i^{(0:T)} | \mathbf{z}) &= p(\mathbf{x}_i^{(T)}) \prod_{t=1}^T p_\theta(\mathbf{x}_i^{(t-1)} | \mathbf{x}_i^{(t)}, \mathbf{z}), \\ p_\theta(\mathbf{x}_i^{(t-1)} | \mathbf{x}_i^{(t)}, \mathbf{z}) &= \mathcal{N}(\mathbf{x}_i^{(t-1)} | \boldsymbol{\mu}_\theta(\mathbf{x}_i^{(t)}, t, \mathbf{z}), \beta_t \mathbf{I}), \end{aligned} \quad (3.2)$$

where the estimated mean $\boldsymbol{\mu}_\theta$ is modeled by a neural network with parameters θ and the generation process is steered by a conditioning latent vector \mathbf{z} . This latent variable is crucial as the generated point cloud needs to have consistent global properties, i.e. realistic energy and center of gravity. As all initial noise points $X^{(T)} = \mathbf{x}_i^{(T)}_{i=1}^N$ are sampled identical and independently distributed (i.i.d.), the latent \mathbf{z} is the only source of shared information for the generated point cloud $X^{(0)}$.

For practical training, ref. [36] showed that $\mathbf{x}_i^{(t)}$ can be sampled directly from data $\mathbf{x}_i^{(0)}$ without going through all previous steps:

$$q(\mathbf{x}_i^{(t)} | \mathbf{x}_i^{(0)}) = \mathcal{N}(\mathbf{x}_i^{(t)} | \sqrt{\bar{\alpha}_t} \mathbf{x}_i^{(0)}, (1 - \bar{\alpha}_t) \mathbf{I}), \quad (3.3)$$

with notation $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^t \alpha_s$. This allows parameterising $\mathbf{x}_i^{(t)}$ as $\mathbf{x}_i^{(t)}(\mathbf{x}_i^{(0)}, \epsilon) = \sqrt{\bar{\alpha}_t} \mathbf{x}_i^{(0)} + \sqrt{1 - \bar{\alpha}_t} \epsilon$ with noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The model approximating the mean μ_θ can be parameterized as:

$$\mu_\theta(\mathbf{x}_i^{(t)}, t) = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_i^{(t)} - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_i^{(t)}, t, \mathbf{z}) \right), \quad (3.4)$$

where $\epsilon_\theta(\mathbf{x}_i^{(t)}, t, \mathbf{z})$ is a neural network which predicts ϵ from $\mathbf{x}_i^{(t)}$. Hence, the model μ_θ was rephrased as a noise predictor ϵ_θ which can be optimized with a simple L2 loss:

$$L_i^{(t-1)} = \mathbb{E}_{\mathbf{x}_i^{(0)}, \epsilon, \mathbf{z}} \left[\left\| \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} \mathbf{x}_i^{(0)} + \sqrt{1 - \bar{\alpha}_t} \epsilon, t, \mathbf{z}) \right\|^2 \right]. \quad (3.5)$$

Therefore the diffusion loss for the full point cloud at time step t is given by $L_{\text{diffusion}}^{(t-1)} = \frac{1}{N} \sum_{i=1}^N L_i^{(t-1)}$. For efficient training, a random time step $t \in 1, \dots, T$ is chosen at each optimization step. A full derivation of the PointWise Net and its training objective can be found in ref. [34]. We optimize our model with a total of $T = 100$ time steps with a quadratic variance scheduler between $\beta_T = 0.02$ and $\beta_1 = 10^{-4}$.

The neural network μ_θ is the same for each point $\mathbf{x}_i^{(t)} \in X^{(t)}$, therefore we can sample permutation-equivariant point clouds with a variable number of points N . Output and input dimensions of the network need to be identical — in our case four (three spatial dimensions and energy). We utilize the same PointWise Net architecture as used in ref. [34]. An overview of the architecture and the layer structure is shown in figure 3. It consists of multiple *ConcatSquash* layers [37] which are conditioned on a global context vector — in this case a concatenated vector of the incident particle energy E , the number of points N , the time features t , and the latent vector \mathbf{z} . The *ConcatSquash* layers are essentially two-layer fully connected networks, with the first hidden layer aligning the dimensionality of the context conditioning vector with the hidden dimensionality of the layer input. A `LEAKYReLU` activation follows the second hidden layer.

This architecture allows for each point to be sampled independently of all other points of the point cloud. Note that the assumption of i.i.d. sampling of each point of the calorimeter shower is theoretically not optimal as in reality, the photon shower cascade develops through particle interactions. However, due to the simple topological structure of electromagnetic showers, we assume the points to be i.i.d. and gain encouraging generative fidelity with this approach. Since we do not model any particle interactions, the point cloud generation scales $\mathcal{O}(N)$ with the number of points, which is crucial since we generate several thousand points per shower. We performed a few early experiments with the equivariant point cloud (EPiC) layers introduced in ref. [28] as they allow for point interactions with linear computational scaling but did not observe improved performance. This is likely due to the EPiC layers being optimised for much smaller point clouds of $\mathcal{O}(10)$ cardinality. Graph network and transformer approaches for point cloud generation such as ref. [29, 38–42] generally scale $\mathcal{O}(N^2)$ and are therefore significantly slower than our model. However, fast implementations of attention, graph networks or sequence convolutions [43–46] might be interesting avenues to explore in the future, e.g. for generating hadronic showers including nuclear interactions.

Figure 4 shows a visualization of the calorimeter point cloud generation at different stages of the reverse diffusion process (i.e. starting with noise).

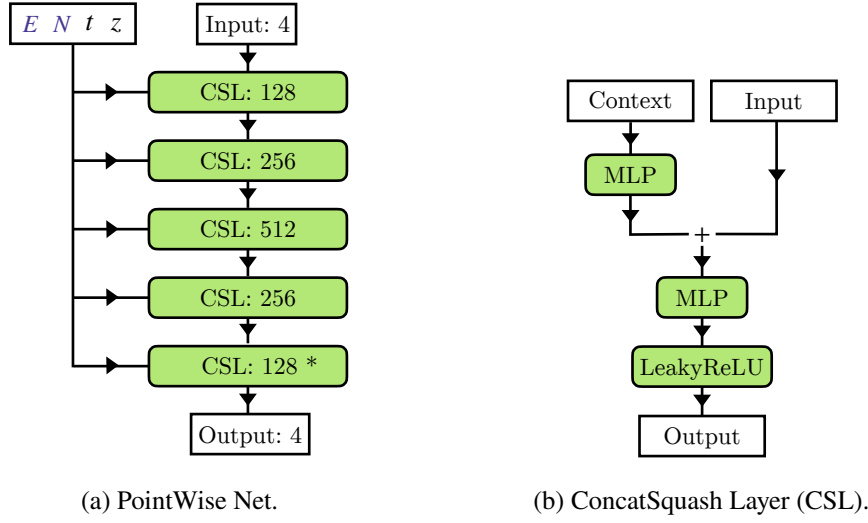


Figure 3. Illustration of the CALOClouds' PointWise Net (a) consisting of multiple ConcatSquash layers (b). The number of hidden dimensions is indicated. MLP denotes a multi-layer perceptron. *No activation function is applied in the last layer of the PointWise Net.

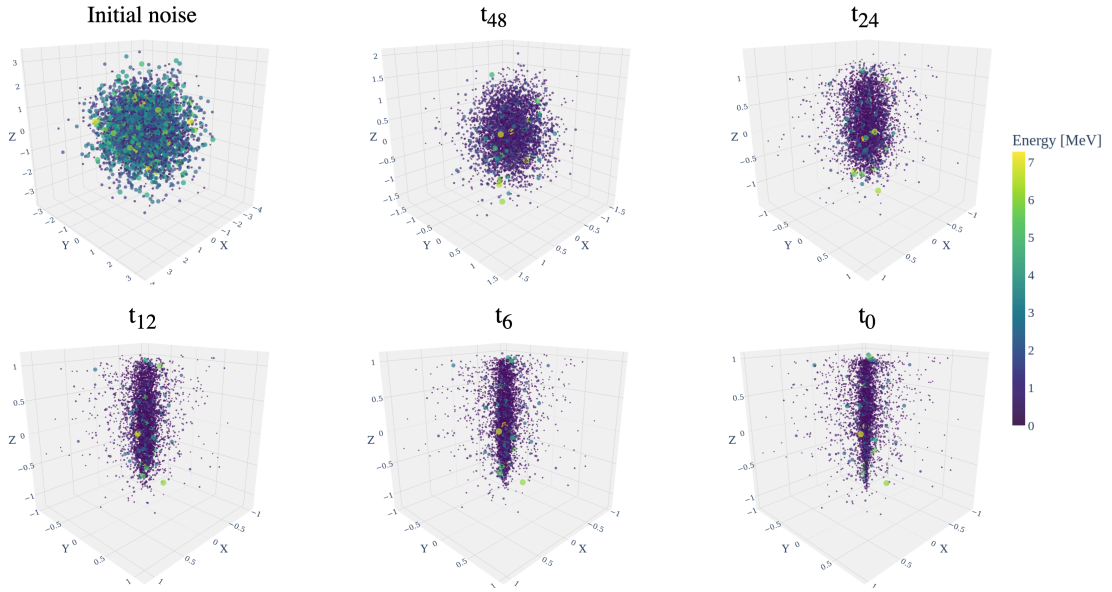


Figure 4. Illustration of the reverse diffusion process. Starting from the initial noise. The color scale corresponds to the point energy.

3.2 Encoder & latent flow

The PointWise Net generates point clouds via the reverse diffusion process. For the resulting point clouds to be realistic, PointWise Net needs to be conditioned on additional quantities beyond multiplicity N and shower energy E .

In principle, one could add additional physically relevant quantities such as the total visible energy, the center of gravity, or the shower start as explicit conditioning features. However, such a

choice of observables might bias the generated showers. Instead, we opt for learning an additional global context vector \mathbf{z} to capture any other relevant distributions via an additional encoder.

This encoding is learned by an Equivariant Point Cloud (EPiC) Encoder using three EPiC layers introduced in ref. [28] with a hidden dimensionality of 128. The EPiC Encoder is conditioned on E and N and learns to encode the original GEANT4 point cloud into two latent space vectors $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. Similar to the encoder in a VAE, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are regularised towards a Gaussian distribution with the Kullback-Leibler divergence (KLD) loss and the latent space \mathbf{z} is sampled with the reparametrization trick [47]. The KLD loss is given by:

$$L_{\text{KLD}} = D_{\text{KL}}(\mathcal{Z}||\mathcal{N}(0, 1)) = -\frac{1}{2} \left(1 + \log(\boldsymbol{\sigma}^2) - \boldsymbol{\mu}^2 - \boldsymbol{\sigma}^2 \right), \quad (3.6)$$

with the latent variables sampled via $\mathbf{z} \sim \mathcal{Z} = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$. We set the size of \mathbf{z} to 256, the default in ref. [34], without performing a hyperparameter optimisation.

During training, the EPiC Encoder and the point cloud generator are trained in parallel using a combination of L_{KLD} and the diffusion reconstruction loss. At a randomly sampled time step t this results in the total loss function:

$$L_{\text{total}}^{(t-1)} = w_{\text{KLD}} \cdot \max(L_{\text{KLD}}, m_{\text{KLD}}) + L_{\text{diffusion}}^{(t-1)}, \quad (3.7)$$

with the KLD weight $w_{\text{KLD}} = 10^{-3}$. To prevent a posterior collapse of \mathbf{z} we enforced a minimum KLD loss of $m_{\text{KLD}} = 1.0$. The EPiC Encoder and the PointWise Net were implemented using PYTORCH [48] and trained together for a total of 800k iterations using the ADAM [49] optimizer with a learning rate scheduled between 10^{-3} and 10^{-4} .

During sampling, we generate the encoded latent space \mathbf{z} with a Normalizing Flow model [50] termed Latent Flow. This Latent Flow model is conditioned on E and N and consists of ten coupling blocks with monotonic rational-quadratic splines [51] each with two layers, a hidden dimensionality of 128, and LEAKYRELU activations. The Latent Flow is trained simultaneously with the EPiC Encoder and the diffusion model via the negative log-likelihood loss but with a separate ADAM optimizer and a learning rate scheduler. It was implemented using the NFLOWS package [52].

A similar encoding and latent flow strategy are implemented in ref. [34], however, we choose a more advanced encoder and flow architecture as well as choosing to disentangle the encoder and diffusion loss from the flow training loss to achieve a more flexible optimization regime.

3.3 Shower flow & post-diffusion calibration

To estimate the number of points N in a point cloud i.e. N_{gen} for a requested incident energy E , we employ a separately trained normalizing flow model, called Shower Flow. This Shower Flow is trained to generate the showers' total visible energy E_{sum} and the number of points per calorimeter layer $N_{z,i}$ for the purpose of post-diffusion calibration. For consistency, the total number of points N_{gen} is defined by $N_{\text{gen}} = \sum_{i=0}^{29} N_{z,i}$.

The Shower Flow consists of ten blocks with seven coupling layers each. Out of these, six are based on affine transformations [53] and one is based on element-wise rational splines [51]. Each coupling layer is conditioned on the incident particle energy E . This flow was implemented with the PYRO package [54].

While the number of points N_{gen} generated by the Shower Flow describes the corresponding GEANT4 distribution N_{data} well, the number of calorimeter hits, i.e. the occupancy O_{gen} , of the projected point cloud may not necessarily correctly match the corresponding GEANT4 distribution O_{data} . This occurs as the projection of the space points into cells of the detector grid has two possible failure modes: either too many points N_{gen} are projected into the same calorimeter cell and therefore the number of hits O_{gen} is lower than the expected O_{data} , or too few points are projected together leading to an overestimation of O_{data} . For our model, we observed the latter case since for a given shower the O_{gen} is up to 10% larger than the O_{data} . To resolve this mismatch, we introduced an energy-dependent correction factor $c_N(N) = p_{\text{gen}}(p_{\text{data}}(N))/N$. Here, p_{data} represents a cubic polynomial fit of N_{data} to O_{data} , while p_{gen} corresponds to a cubic polynomial fit of O_{gen} to N_{gen} . This correction factor is used to rescale N_{gen} predicted by the Shower Flow as well as the predicted points per layer $N_{z,i}$, resulting in a calibrated number of points $N_{\text{cal}} = c_N \cdot N_{\text{gen}}$ and $N_{z,i,\text{cal}} = c_N \cdot N_{z,i}$. The calibrated number of points is then used for sampling from the diffusion model.

Finally, we consider an additional rescaling of the point features following the diffusion models' generation: the post-diffusion calibration is performed in four steps. First, in case negative point energies were generated, they are set to zero. Second, the total energy of the point cloud is rescaled to match the predicted E_{sum} . Third, all points are ordered based on their Z-coordinate and then iteratively the first $N_{z,i=0}$ points are set to Z-coordinate index $z_i = 0$, afterward the next $N_{z,i=1}$ points are set to coordinate $z_i = 1$ and so on until layer index $z_i = 29$. This way, the total energy and the energy distribution in the Z-direction are well calibrated, otherwise both distributions are challenging to model with the diffusion model alone precisely. Fourth, a calibration of the center of gravity in the X- and Y-directions is performed by shifting all X- and Y-coordinates by their mean center of gravity of the training data set.

Overall the CALOCLOUDS model together with these calibrations achieves high fidelity on a number of important shower physics observables as presented in the following section.

4 Results

In the following section 4.1, we compare the distributions produced by the CALOCLOUD model to the GEANT4 ground-truth for several per-shower variables and global observables. For this, the space points generated with CALOCLOUD are projected back into the real and irregular grid-like geometry of the cells in the ILD detector model — in the same manner as in the GEANT4 simulation. Section 4.2 then demonstrates the model's ability to generate showers originating from any arbitrary incident point of the particle along the detector surface. Finally, in section 4.3 we conclude with an investigation of our model's speed-up in comparison to GEANT4 simulation.

4.1 Physics performance

First, we consider how well the distribution of measured energies across a broad range of incident energies is described after projecting back to the physical coordinate system. Figure 5 (left) shows the per-cell energy distribution. Overall, this is well described by the diffusion model, with the largest differences occurring at regions in the spectrum where the slope changes (i.e. at 0.5 MeV and 100 MeV). Here, the shaded region indicates energies below 0.1 MeV, corresponding to half the energy deposited by a minimally ionizing particle (MIP). As these energies will be dominated by

detector noise they are ignored in the reconstruction process and therefore are also not considered when calculating other quantities.

The radial shower profile (figure 5 (center)) and the longitudinal shower profile (figure 5 (right)) are overall well-learned by the diffusion model with some differences observed at the edges of the distribution.

Next, we consider the modeling of the center of the shower in figure 6. For this, the center of gravity (i.e. the energy-weighted centroid) is calculated per shower along the X (left), Y (center), and Z (right) direction. The radial directions in X and Y are well described, with the diffusion model predicting a slightly more peaked distribution in both cases. In the longitudinal direction, the shape is captured correctly by the diffusion model as well, with the peak shifted approximately 10 mm compared to GEANT4.

To test whether the response for different shower energies is learned correctly, we consider the visible energy (left) and the overall number of hits above the threshold (right) in figure 7. The distributions are shown for incident energies of 10, 50, and 90 GeV. At all incident energies, the visible energy distributions are very well modeled — an important feature for a calorimeter simulation as it is the width of these distributions that directly determines the energy resolution of the detector. The distributions for the number of hits above the threshold are modeled well with, some deviations visible at higher energies.

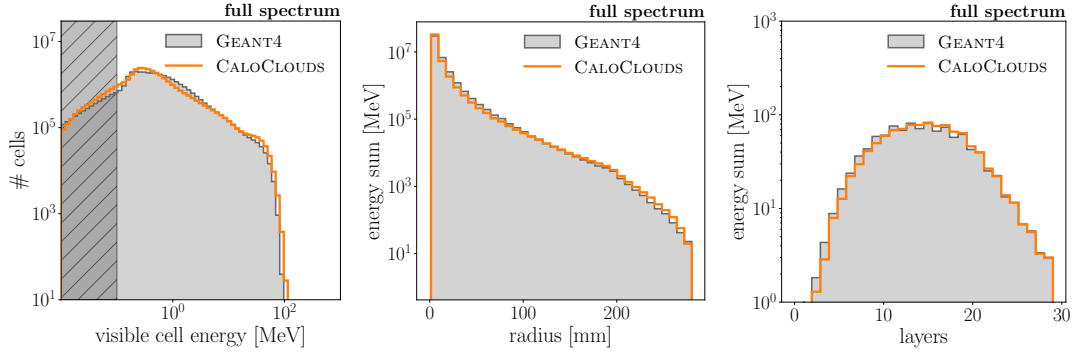


Figure 5. Histograms of the cell energies (left), radial shower profile (center), and longitudinal shower profile (right) for both GEANT4 and CALOCLOUDS. In the per-cell energy distribution, the region below 0.1 MeV is grayed out (see main text for details). All distributions are calculated for a uniform distribution of incident particle energies.

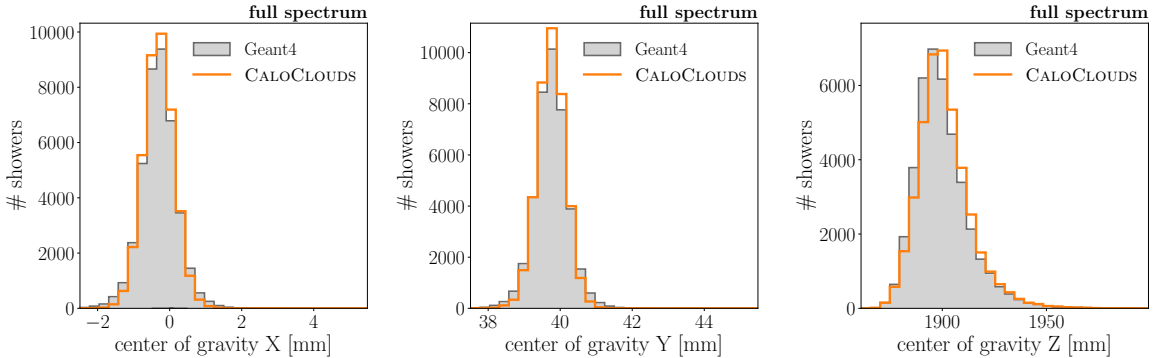


Figure 6. Position of the center of gravity of showers along the X (left), Y (center), and Z (right) directions. All distributions are calculated for a uniform distribution of incident particle energies.

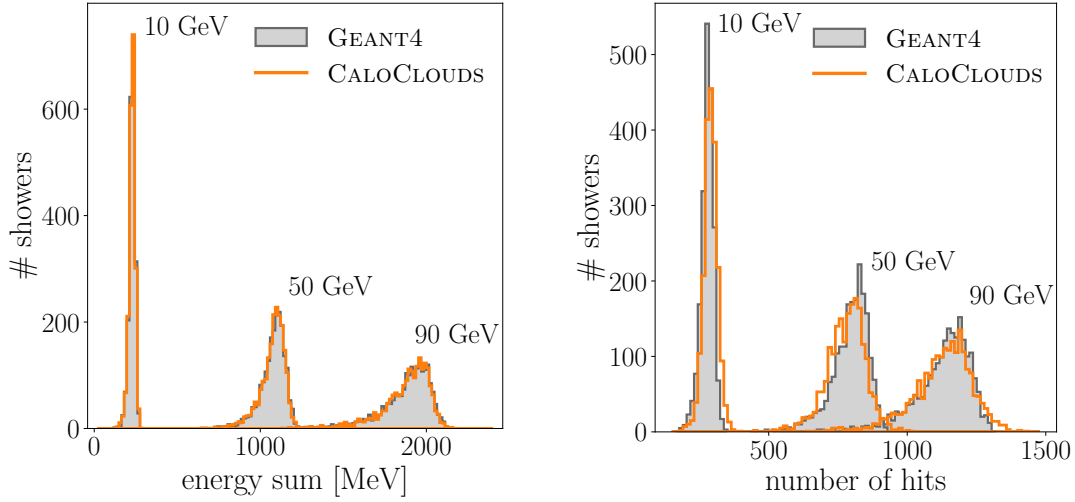


Figure 7. Visible energy sum (left) and the number of hits (right) distributions, for 10, 50, and 90 GeV showers.

4.2 Shower translation

To test the capability of the CALOCLOUDS architecture to generate photon showers originating from arbitrary incident points on the detector, we compare the model using the two different data sets introduced in section 2: a validation data set that has been simulated at the same position in the calorimeter as the training data set and a test data set simulated at a different position in the calorimeter. Depending on the exact impact point in the calorimeter the local cell geometry will look different, for example, due to the local staggering of cells between layers, the position of the impact point with respect to the cell centers, gaps in the cell structure at the edge of silicon sensors, etc. The distribution that is most susceptible to potential artifacts resulting from such a translation is the cell energy distribution. This distribution is shown in figure 8 for the validation (left) and test data set (right) respectively, where in the case of the test data set the generated point cloud has been translated to the new impact position and then projected into the local cell geometry used for the full simulation with GEANT4. Both comparisons show equally reasonable agreement between GEANT4 and CALOCLOUDS with no visible deterioration for the translated point cloud.

4.3 Timing

The main motivation for simulation using generative models is speeding up the generation time per shower. In table 2, we present the generation time of CALOCLOUDS in comparison with the baseline GEANT4 simulation on either a single CPU core or on a machine equipped with an NVIDIA A100 GPU. The GEANT4 baseline time is taken from ref. [16], which was obtained in an identical detector model and sub-detector to that studied here. In all cases, we consider uniform energy distributions with incident particle energies between 10 and 100 GeV. Note that the range of 10–100 GeV for timing studies is chosen for consistency with ref. [16] but exceeds the training range of 10–90 GeV used for the CALOCLOUDS model. However, while this might in principle be out-of-distribution for modeling the data well, we expect the timing performance to be robust also in extrapolation.

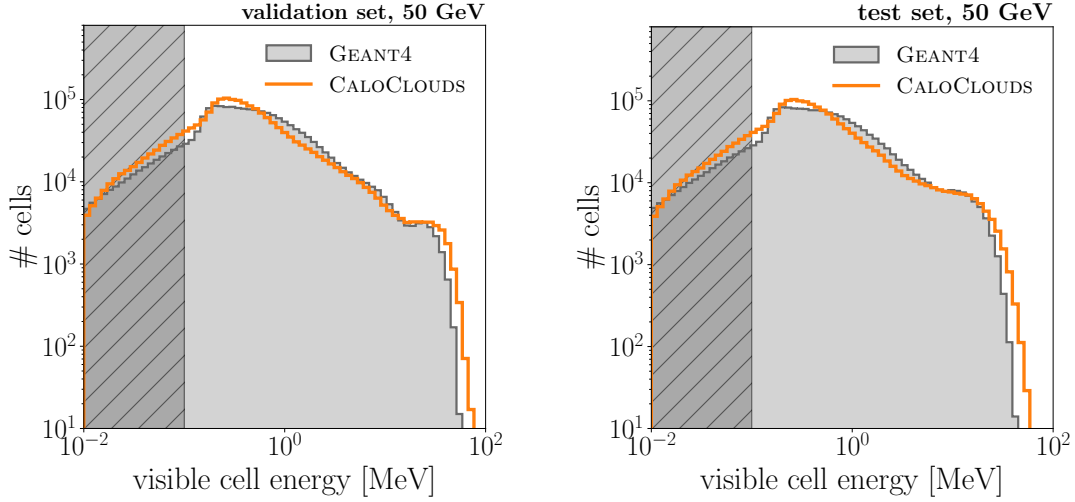


Figure 8. Per-cell energy distribution for the 50 GeV validation (left) data set, created at the same position as the training data set and for a 50 GeV test (right) data set simulated at a different position with the generated point cloud translated to this position.

Table 2. Comparison of the computational performance of CALOCLOUDS to the baseline GEANT4 simulator on a single core of an Intel® Xeon® CPU E5-2640 v4 (CPU) and an NVIDIA® A100 with 40 GB of memory (GPU). Showers were generated with incident energy uniformly distributed between 10 and 100 GeV. The batch size was set to 1 on the CPU and to 64 on the GPU. Values presented are the means and standard deviations over 25 runs. The GEANT4 time is taken from ref. [16].

Hardware	Simulator	Time / Shower [ms]	Speed-up
CPU	GEANT4	4082 ± 170	$\times 1$
	CALOCLOUDS	3509 ± 220	$\times 1.2$
GPU	CALOCLOUDS	38 ± 3	$\times 107$

On a single CPU core, the speed-up provided by CALOCLOUDS is about 1.2 \times , whereas on a GPU the relative speed-up is increased to 107 \times faster than GEANT4.

When sampling, the Shower Flow is evaluated once for all events, followed by batch-wise sampling from the Latent flow and the PointWise Net. Unlike previous image-based fast simulation models such as ref. [16, 18, 21], the computational cost of the PointWise Net scales $\mathcal{O}(N)$ with the number of points and hence with the energy E just like GEANT4. Therefore, to improve overall training and sampling speed, we batch together events with a similar number of points. Overall, the model is trained for 800k iterations on an NVIDIA A100, which takes about 80 hours.

Our implementation of PointWise Net uses a 100-step reverse diffusion process for shower generation. This reverse diffusion process is implemented as a computational loop and can therefore not be parallelized. Recent developments investigated how few-step or even single-step generation

with diffusion models might be possible, e.g. using consistency distillation [55] or progressive distillation [56], which has been already applied in high-energy physics [23]. Using one of these techniques will likely speed up the generation by at least 10× since the diffusion process in our model is significantly slower than the other components such as the Shower and Latent Flow generation. Since this work is focused on model development, such improvements are left to future work.

5 Conclusions

Motivated by the overarching need of simulating particle showers in complex calorimeter detectors in a time and resource-efficient manner, generative models are widely explored in current particle physics research. A key problem is how to scale up such generators to be able to simulate showers with thousands of particles in high-resolution calorimeter regions. Given the inefficient scaling behavior of fixed-structure approaches, point clouds are an attractive alternative and form the foundation of our proposed strategy.

This work makes several contributions to advance the state-of-the-art in this task. At the core of the proposed CALOCLOUDS architecture is the PointWise Net, a permutation invariant, diffusion-based, point cloud generator. It is accompanied by an EPiC Encoder (in the training phase), a Latent Flow (in the sampling phase) to provide realistic conditions, and an additional Shower Flow that learns the visible energy distribution and the number of points per layer.

In principle, this combination of four models could already be used to directly simulate a point cloud of calorimeter hits. However, the relatively dense arrangement of hits is difficult to model, with the correct number of hits per volume, in particular, being difficult to learn. Additional difficulties arise from realistic calorimeter cell layouts due to staggering between layers or gaps between sensors, resulting in different local cell topologies in different parts of the calorimeter. In order to circumvent these constraints we, for the first time, move to an even higher-resolution space and consider individual GEANT4 steps. While this increases the number of points to simulate by a small integer factor, it yields a distribution that can be more easily learned and subsequently downsampled to the actual resolution of a given calorimeter.

Together these two innovations allow us, for the first time, to generate high-cardinality point clouds with unprecedented fidelity in all observed distributions with moderate speed-ups compared to GEANT4. In follow-up studies, techniques such as consistency distillation [55] have been proven to significantly increase the sampling speed [57]. Additional comparisons of the CALOCLOUDS model to already established fixed-structure calorimeter generative models, i.e. the BIB-AE [16] and CaloScore [22], we leave to further research. Based on the shown results, these models currently offer higher generative fidelity, yet are geometry-dependent and suffer from the discussed computational inefficiencies. For future research, we plan to improve the CALOCLOUDS diffusion paradigm and the model architecture, taking inter-point correlations into account, to increase the generative fidelity.

For scenarios where higher resolution versions of simulations are not available for training (e.g. shower simulations, samples taken from collider data, the ongoing *CaloChallenge* [58] competition or problems from other domains), it might be possible to generate surrogate high-resolution data by smearing points with an appropriate function. Even so, CALOCLOUDS demonstrates the potential of scaling point cloud simulations with high fidelity to much higher cardinalities, making the realistic simulation of showers at the HL-LHC, the ILC, and beyond a feasible target.

Acknowledgments

We thank Sarah Heim for the valuable comments on the manuscript. This research was supported in part by the Maxwell computational resources operated at Deutsches Elektronen-Synchrotron DESY, Hamburg, Germany. This project has received funding from the European Union’s Horizon 2020 Research and Innovation programme under Grant Agreement No 101004761. We acknowledge support by the Deutsche Forschungsgemeinschaft under Germany’s Excellence Strategy — EXC 2121 Quantum Universe — 390833306 and via the KISS consortium (05D23GU4, 13D22CH5) funded by the German Federal Ministry of Education and Research BMBF in the ErUM-Data action plan. E.B. is partially funded by a scholarship from the Friedrich Naumann Foundation for Freedom. E.B. and W.K. acknowledge funding by the German Federal Ministry of Science and Research (BMBF) via *Verbundprojekts 05H2018 - R&D COMPUTING (Pilotmaßnahme ErUM-Data) Innovative Digitale Technologien für die Erforschung von Universum und Materie*. A.K. has received support from the Helmholtz Initiative and Networking Fund’s initiative for refugees as a refugee of the war in Ukraine.

A Effects of the pre-clustering

In order to study possible effects of the pre-clustering procedure (see section 2), we test its closure by applying the procedure to a sample simulated with GEANT4 alone and then project the resulting clusters that form the point cloud used in CALOCLOUD back into the realistic cell geometry of the ILD simulation model. Figure 9 shows this for a typical shower with 90 GeV incident energy in a given layer of the calorimeter (layer 21). The left figure shows the cell energies in this layer as generated during the GEANT4 simulation by projecting the energy depositions into the calorimeter cells. The thick black bars are gaps between sensors as they occur in the simulation model of ILD at the chosen impact point. The center plot shows the resulting space points after the projection of the GEANT4 steps into the virtual ultra-high granular grid. Finally, the right plot shows the cell energies resulting from simply projecting the space points back into the calorimeter cells. Additionally, the GEANT4 steps are shown on the left in blue and the clusters on the right in orange. No obvious discrepancies are seen in this example.

In order to quantify the effects, in figure 10 we show an overlay of 2,000 showers in the same layer. The left figure shows this overlay as produced by GEANT4, while the center plot shows this overlay after a full round trip of pre-clustering and projecting back into the same cell geometry. The right plot shows the resulting relative difference for every cell. The resulting difference is well below 10% for all and below 2% for most cells, indicating good closure. Importantly, most differences occur in sparsely populated parts of the showers where small differences in absolute energies lead to large relative differences. The core, where most of the shower energy is concentrated, is especially well preserved.

Finally, figure 11 presents a comparison of this difference plot for the applied procedure with 36 times higher granularity (left — same as figure 10-right), as used in this paper, and a difference plot arising from a granularity increased by only 4 times (left). While such a lower granularity would potentially result in faster sampling times, it also shows significantly larger differences compared to the applied pre-clustering procedure.

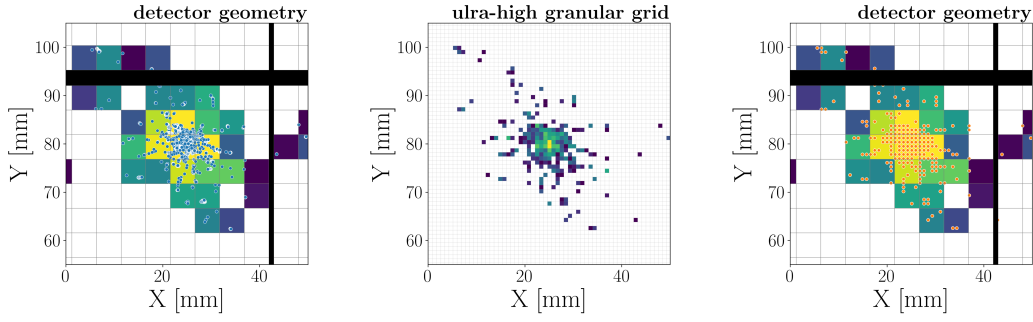


Figure 9. Example shower of 90 GeV in one layer of the calorimeter. Left: cell energies as recorded in the ILD simulation model with blue dots representing the GEANT4 steps. Center: projection of the steps in the ultra-high granularity grid (pre-clustering). Right: resulting cell energies after projecting the space points (orange dots) back into the detector cells.

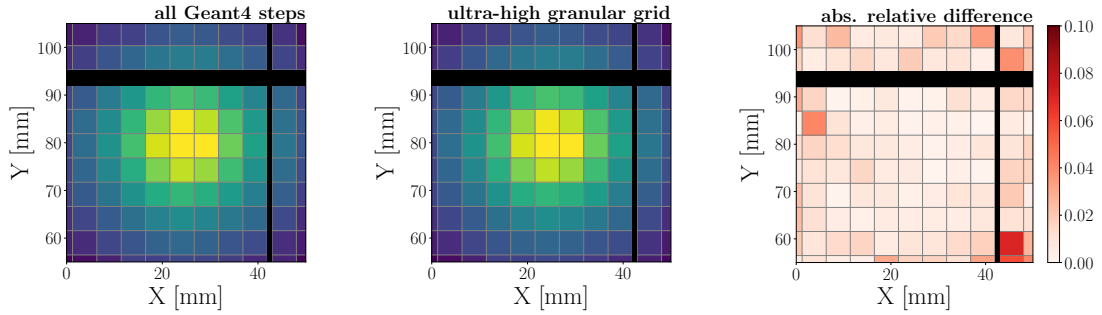


Figure 10. Overlay of 2000 90 GeV showers in one calorimeter layer. Left: as simulated in GEANT4. Center: same showers after a full round trip of pre-clustering and projecting back into the detector cells. Right: relative difference of the initial and processed distribution.

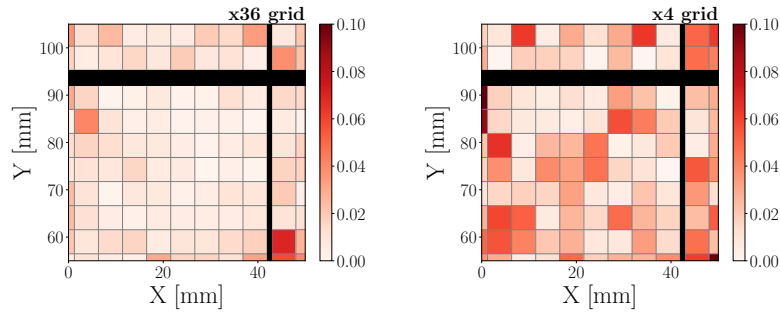


Figure 11. Difference after pre-clustering and projecting back into the cell geometry for 36 times (left) and 4 times (right) increased granularity.

References

- [1] HEP SOFTWARE FOUNDATION collaboration, *A Roadmap for HEP Software and Computing R&D for the 2020s*, *Comput. Softw. Big Sci.* **3** (2019) 7 [[arXiv:1712.06982](https://arxiv.org/abs/1712.06982)].

- [2] A. Boehnlein et al., *HL-LHC Software and Computing Review Panel Report*, Tech. Rep. [CERN-LHCC-2022-007](#), CERN, Geneva (2022).
- [3] ATLAS collaboration, *The ATLAS Simulation Infrastructure*, *Eur. Phys. J. C* **70** (2010) 823 [[arXiv:1005.4568](#)].
- [4] A. Butter et al., *GANplifying event samples*, *SciPost Phys.* **10** (2021) 139 [[arXiv:2008.06545](#)].
- [5] S. Bieringer et al., *Calomplification — the power of generative calorimeter models*, *2022 JINST* **17** P09028 [[arXiv:2202.07352](#)].
- [6] M. Paganini, L. de Oliveira and B. Nachman, *Accelerating Science with Generative Adversarial Networks: An Application to 3D Particle Showers in Multilayer Calorimeters*, *Phys. Rev. Lett.* **120** (2018) 042003 [[arXiv:1705.02355](#)].
- [7] M. Paganini, L. de Oliveira and B. Nachman, *CaloGAN: Simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks*, *Phys. Rev. D* **97** (2018) 014021 [[arXiv:1712.10321](#)].
- [8] L. de Oliveira, M. Paganini and B. Nachman, *Controlling Physical Attributes in GAN-Accelerated Simulation of Electromagnetic Calorimeters*, *J. Phys. Conf. Ser.* **1085** (2018) 042017 [[arXiv:1711.08813](#)].
- [9] M. Erdmann, L. Geiger, J. Glombitza and D. Schmidt, *Generating and refining particle detector simulations using the Wasserstein distance in adversarial networks*, *Comput. Softw. Big Sci.* **2** (2018) 4 [[arXiv:1802.03325](#)].
- [10] M. Erdmann, J. Glombitza and T. Quast, *Precise simulation of electromagnetic calorimeter showers using a Wasserstein Generative Adversarial Network*, *Comput. Softw. Big Sci.* **3** (2019) 4 [[arXiv:1807.01954](#)].
- [11] S. Vallecorsa, *Generative models for fast simulation*, *J. Phys. Conf. Ser.* **1085** (2018) 022005.
- [12] F. Carminati et al., *Three dimensional Generative Adversarial Networks for fast simulation*, *J. Phys. Conf. Ser.* **1085** (2018) 032016.
- [13] P. Musella and F. Pandolfi, *Fast and Accurate Simulation of Particle Detectors Using Generative Adversarial Networks*, *Comput. Softw. Big Sci.* **2** (2018) 8 [[arXiv:1805.00850](#)].
- [14] ATLAS collaboration, *Deep generative models for fast shower simulation in ATLAS*, Tech. Rep. [ATL-SOFT-PUB-2018-001](#), CERN, Geneva (2018).
- [15] H. Hashemi et al., *Ultra-High-Resolution Detector Simulation with Intra-Event Aware GAN and Self-Supervised Relational Reasoning*, [arXiv:2303.08046](#).
- [16] E. Buhmann et al., *Getting High: High Fidelity Simulation of High Granularity Calorimeters with High Speed*, *Comput. Softw. Big Sci.* **5** (2021) 13 [[arXiv:2005.05334](#)].
- [17] E. Buhmann et al., *Hadrons, better, faster, stronger*, *Mach. Learn. Sci. Tech.* **3** (2022) 025014 [[arXiv:2112.09709](#)].
- [18] S. Diefenbacher et al., *New angles on fast calorimeter shower simulation*, *Mach. Learn. Sci. Tech.* **4** (2023) 035044 [[arXiv:2303.18150](#)].
- [19] C. Krause and D. Shih, *Fast and accurate simulations of calorimeter showers with normalizing flows*, *Phys. Rev. D* **107** (2023) 113003 [[arXiv:2106.05285](#)].
- [20] C. Krause and D. Shih, *Accelerating accurate simulations of calorimeter showers with normalizing flows and probability density distillation*, *Phys. Rev. D* **107** (2023) 113004 [[arXiv:2110.11377](#)].

- [21] S. Diefenbacher et al., *L2LFlows: generating high-fidelity 3D calorimeter images*, *2023 JINST* **18** P10017 [[arXiv:2302.11594](#)].
- [22] V. Mikuni and B. Nachman, *Score-based generative models for calorimeter shower simulation*, *Phys. Rev. D* **106** (2022) 092009 [[arXiv:2206.11898](#)].
- [23] V. Mikuni, B. Nachman and M. Pettee, *Fast point cloud generation with diffusion models in high energy physics*, *Phys. Rev. D* **108** (2023) 036025 [[arXiv:2304.01266](#)].
- [24] M. Leigh et al., *PC-JeDi: Diffusion for Particle Cloud Generation in High Energy Physics*, [arXiv:2303.05376](#).
- [25] ATLAS collaboration, *AtlFast3: The Next Generation of Fast Simulation in ATLAS*, *Comput. Softw. Big Sci.* **6** (2022) 7 [[arXiv:2109.02551](#)].
- [26] A. Martelli, *The CMS HGCal detector for HL-LHC upgrade*, in the proceedings of the 5th Large Hadron Collider Physics Conference, Shanghai, China, 15–20 May 2017 [[arXiv:1708.08234](#)].
- [27] T. Behnke et al., eds., *The International Linear Collider Technical Design Report — Volume 1: Executive Summary*, [arXiv:1306.6327](#).
- [28] E. Buhmann, G. Kasieczka and J. Thaler, *EPiC-GAN: Equivariant Point Cloud Generation for Particle Jets*, *SciPost Phys.* **15** (2023) 130 [[arXiv:2301.08128](#)].
- [29] R. Kansal et al., *Particle Cloud Generation with Message Passing Generative Adversarial Networks*, in the proceedings of the 35th Conference on Neural Information Processing Systems, Online Conference, Canada, 6–14 December 2021 [[arXiv:2106.11535](#)].
- [30] ILD CONCEPT GROUP collaboration, *International Large Detector: Interim Design Report*, [arXiv:2003.01116](#).
- [31] The iLCSoft Group, *iLCSoft Project Page*, <https://github.com/iLCSoft>.
- [32] M. Frank, F. Gaede, C. Grefe and P. Mato, *DD4hep: A Detector Description Toolkit for High Energy Physics Experiments*, *J. Phys. Conf. Ser.* **513** (2014) 022010.
- [33] GEANT4 collaboration, *GEANT4 — a simulation toolkit*, *Nucl. Instrum. Meth. A* **506** (2003) 250.
- [34] S. Luo and W. Hu, *Diffusion Probabilistic Models for 3D Point Cloud Generation*, [arXiv:2103.01458](#).
- [35] J. Sohl-Dickstein, E.A. Weiss, N. Maheswaranathan and S. Ganguli, *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*, [arXiv:1503.03585](#).
- [36] J. Ho, A. Jain and P. Abbeel, *Denoising Diffusion Probabilistic Models*, [arXiv:2006.11239](#).
- [37] W. Grathwohl et al., *FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models*, [arXiv:1810.01367](#).
- [38] J. Lee et al., *Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks*, [arXiv:1810.00825](#).
- [39] A.R. Kosiorek, H. Kim and D.J. Rezende, *Conditional Set Generation with Transformers*, [arXiv:2006.16841](#).
- [40] J. Kim, J. Yoo, J. Lee and S. Hong, *SetVAE: Learning Hierarchical Composition for Generative Modeling of Set-Structured Data*, [arXiv:2103.15619](#).
- [41] F.A. Di Bello et al., *Conditional Generative Modelling of Reconstructed Particles at Collider Experiments*, [arXiv:2211.06406](#).

- [42] R. Kansal et al., *Evaluating generative models in high energy physics*, *Phys. Rev. D* **107** (2023) 076017 [[arXiv:2211.10295](#)].
- [43] S.R. Qasim, J. Kieseler, Y. Iiyama and M. Pierini, *Learning representations of irregular particle-detector geometry with distance-weighted graph networks*, *Eur. Phys. J. C* **79** (2019) 608 [[arXiv:1902.07987](#)].
- [44] T. Dao et al., *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness*, [arXiv:2205.14135](#).
- [45] A. Gu, K. Goel and C. Ré, *Efficiently Modeling Long Sequences with Structured State Spaces*, [arXiv:2111.00396](#).
- [46] M. Poli et al., *Hyena Hierarchy: Towards Larger Convolutional Language Models*, [arXiv:2302.10866](#).
- [47] D.P. Kingma and M. Welling, *Auto-Encoding Variational Bayes*, [arXiv:1312.6114](#).
- [48] A. Paszke et al., *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, [arXiv:1912.01703](#) [[DOI:10.48550/ARXIV.1912.01703](#)].
- [49] D.P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, [arXiv:1412.6980](#).
- [50] D.J. Rezende and S. Mohamed, *Variational Inference with Normalizing Flows*, [arXiv:1505.05770](#).
- [51] C. Durkan, A. Bekasov, I. Murray and G. Papamakarios, *Neural Spline Flows*, [arXiv:1906.04032](#).
- [52] C. Durkan, A. Bekasov, I. Murray and G. Papamakarios, *nflows: normalizing flows in PyTorch*, <https://zenodo.org/record/4296287> [[DOI:10.5281/ZENODO.4296287](#)].
- [53] L. Dinh, J. Sohl-Dickstein and S. Bengio, *Density estimation using Real NVP*, [arXiv:1605.08803](#).
- [54] E. Bingham et al., *Pyro: Deep Universal Probabilistic Programming*, [arXiv:1810.09538](#).
- [55] Y. Song, P. Dhariwal, M. Chen and I. Sutskever, *Consistency Models*, [arXiv:2303.01469](#).
- [56] T. Salimans and J. Ho, *Progressive Distillation for Fast Sampling of Diffusion Models*, [arXiv:2202.00512](#).
- [57] E. Buhmann et al., *CaloClouds II: Ultra-Fast Geometry-Independent Highly-Granular Calorimeter Simulation*, [arXiv:2309.05704](#).
- [58] M.F. Giannelli, G. Kasieczka, C. Krause, D.S. Ben Nachman, D. Shih and A. Zaborowska, *Fast Calorimeter Simulation Challenge 2022*, <https://calochallenge.github.io/homepage/>.