

# Physics-aware modelling of an accelerated particle cloud

Emmanuel Goutierre, Christelle Bruni, Johanne Cohen, Hayg Guler, Michèle Sebag

▶ **To cite this version:**

Emmanuel Goutierre, Christelle Bruni, Johanne Cohen, Hayg Guler, Michèle Sebag. Physics-aware modelling of an accelerated particle cloud. MLPS 2023 - Machine Learning and the Physical Sciences Workshop 23023 - At the 37th conference on Neural Information Processing Systems (NeurIPS), Dec 2023, New Orleans, United States. hal-04396175

**HAL Id: hal-04396175**
**https://hal.science/hal-04396175v1**

Submitted on 16 Jan 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons CC0 1.0 - Universal - International License

# Physics-aware Modeling of an Accelerated Particle Cloud

**Emmanuel Goutierre**
Université Paris-Saclay, CNRS
LISN, 91190 Gif-sur-Yvette, France
IJCLab, 91405 Orsay, France
emmanuel.goutierre@upsaclay.fr

**Christelle Bruni**
Université Paris-Saclay, CNRS
IJCLab, 91405 Orsay, France

**Johanne Cohen**
Université Paris-Saclay, CNRS
LISN, 91190 Gif-sur-Yvette, France

**Hayg Guler**
Université Paris-Saclay, CNRS
IJCLab, 91405 Orsay, France

**Michèle Sebag**
Université Paris-Saclay, CNRS, INRIA
LISN, 91190 Gif-sur-Yvette, France

## Abstract

Particle accelerator simulators, pivotal for acceleration optimization, are computationally heavy; surrogate, machine learning-based models are thus trained to facilitate the accelerator fine-tuning. While these current models are efficient, they do not allow for simulating the beam at the individual particle-level. This paper adapts point cloud deep learning methods, developed for computer vision, to model particle beams.

## 1 Introduction

In the field of particle accelerator physics, simulators are crucial for optimizing the acceleration process. These simulators generate (macro-)particles and track their trajectories through the accelerator. They provide data on various parameters such as the beam barycenter, size, charge and emittance of beams. They can provide all information for the simulated particles to support in-depth analysis. Monitoring this information is fundamental for effectively controlling particle beams during acceleration. However, a notable drawback of existing simulators is that they are computationally heavy, often limiting their extensive usage.

Machine learning (ML) surrogate models emerge as a promising solution to address the computational challenges encountered with traditional simulators in particle accelerator physics. These models efficiently generate statistical data about beam properties [4]. However, a significant gap in their functionalities is that they do not provide detailed information about the individual particles within the beams, which is only delivered through tracking particles in traditional simulators. This gap hinders the full grasp of the beam 6D phase space with ML-based models.

This paper is thus devoted to enhancing these surrogate models, allowing them to handle individual particle data. The proposed approach builds upon advancements in 3D computer vision, paving the way to efficient handling of 2 or 3D point clouds. Specifically, PointNet [12] involves a specific architecture for processing point sets, using a shared network to extract features from individual points and employing a symmetric function to aggregate these features into a global representation of the set. Considering that a particle beam can be represented as an 8D point set comprising three positional

Machine Learning and the Physical Sciences Workshop, NeurIPS 2023.

dimensions, three momentum dimensions, intrinsic time, and charge, it aligns naturally with the capabilities of PointNet. [1] Overall, this paper presents the adaptation of PointNet for regression tasks targeting particle coordinates within the beam.

## 2 Related Works

**Surrogate models of particle accelerators.** Surrogate models offer a streamlined approach to emulating particle accelerator behaviors without the associated computational burden of direct simulations. Neural network-based surrogates present a viable path to creating rapid and reliable accelerator models. [4, 9, 8] utilize fully connected neural networks to forecast simulator outputs based on the accelerator control parameters.

Polynomial chaos expansion, as highlighted in [1], aids in creating surrogate models for cyclotrons, focusing on evaluating machine sensitivities around specific control settings. Such models have been instrumental in enhancing multi-objective optimization processes [4]. By incorporating deconvolutional layers post feed-forward stages, [5] projects the Longitudinal Phase Space of beams, while [14] introduces a convolutional layer for initial laser distribution representation. Further advancements include incorporating deep and invertible neural networks as surrogates, which are efficient alternatives to computationally intensive physics-based models [2].

**Point cloud representation.** As an electron beam essentially is a set of 8D points, its representation can benefit from advances in computer vision and computer graphic applications. Various methods are developed to capture and analyze the spatial information contained within point clouds.

The first option is based on the voxelization of the space [19], reporting the particle density and other average specifics in each voxel. This fixed parametric representation suffers from the tradeoff between the precision $\tau$ and the number of voxels in $\tau^8$.

The second option pioneered by [16] is based on multiple 2d-projections of the point cloud, processed using convolutional architectures and concatenated to define an embedding of the distribution.

The third option, introduced by PointNet [12], relies on direct processing of the points within the point cloud. Given a set of particles $\{x_i\}$in $\mathbb{R}^d$ sampled after distribution $\mathcal{D}$, PointNet computes $\mathcal{D}_{\#f} = \{f(x_i)\}$ with $f$ a mapping from $\mathbb{R}^d$ to $\mathbb{R}^{d'}$, and returns an aggregation of $\mathcal{D}_{\#f}$ through a symmetric function $g$. The merit of the approach is that aggregation function $g$ is learned and can adapt to the varying densities and scales of the data. The versatility of this architecture inspires numerous development of applications in computer vision, from object classification to semantic segmentation [13, 20, 18].

## 3 The LinacNet approach

The proposed approach emulates the physical structure of the accelerator, involving a sequence of $N = 25$ segments. Each segment operates on the beam, a set of particles emitted by the former segment, and propagates it according to its control parameters (settings of the various electromagnetic elements attached to the segment), aimed at specific purposes (e.g. controlling the deviation or the diameter of the beam) through diverse physic mechanisms. In the following, $D_i = \{(\mathbf{x}_{i,j}, y_{i,j}), j = 1 \dots n\}$ denotes the (ground truth) set of $n$ particles forming the input beam of the $i$-th segment, with $\mathbf{x}_{i,j}$ in $\mathcal{R}^8$ describing the position and speed of the $j$-th particle and $y_{i,j}$ in $\{0, 1\}$ stating whether the particle is present or has been deviated before reaching segment $i$. The number $n$ of particles is set to 10,000 in the experiments.

**Architectures.** According to physics [17], the propagation of each particle depends on the whole set of particles. For this reason, the propagation effectuated by the $i$-th segment is modeled using a PointNet architecture. Each layer uses the GELU activation function.

Two PointNet architectures are compared (Fig. 1): in LINACNET.(a), the control settings are part of the input of the PointNet module, whereas in LINACNET.(b), they are concatenated to the output of

[1] Graph Neural Networks (GNN) constitute another way of representing interacting particles. After preliminary experiments, however, full GNN shows that beam simulator modelling is too computationally expensive. A partial GNN architecture is considered in Section 3.

the PointNet module. The question is whether PointNet better exploits the control settings to form the aggregated representation of the beam or is better used to augment the aggregated representation.

The PointNet's max operator is replaced by a weighted average operator to reflect better the overall distribution, where the particle weight is set to the probability of its presence.

Two baseline architectures are also considered to assess the importance of particle interactions. The first baseline architecture is a Siamese network [3], involving a (single, shared) MLP that predicts each particle coordinates at a segment's end based on its initial coordinates and the segment control settings.

The second baseline architecture is GNN-like, where the aggregation operation is performed on $k$ local clusters. Each cluster is associated with a (uniformly sampled) particle referred to as a seed, and particles are related to the cluster of their nearest seed. The local information of each particle $f(x_i)$ is concatenated with the aggregation of the information of its cluster before the last regression module. In our experiments, the number $k$ of clusters is set to 100 ($1\%$ of the number $n$ of particles).

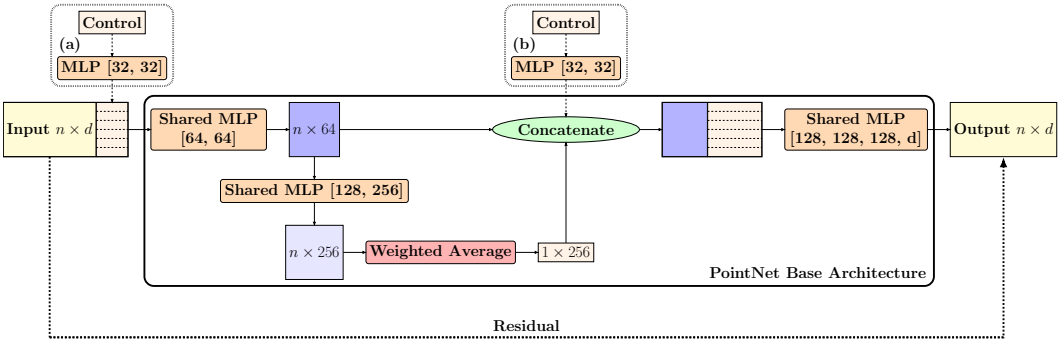



Figure 1: LINACNET Architecture. In LINACNET.(a), the local control settings are provided as input of the particles processed by PointNet. In LINACNET.(b), the control settings is concatenated to the output of the PointNet. Each layer is followed by a GELU [7] activation function.

**Losses and training strategies.** As the whole accelerator is composed of a sequence of segments, the global architecture of LINACNET is made of the composition of neural nets emulating segments 0 to $N-1$. The loss associated with the $i$-th neural net includes the mean square error for every particle (MSE: $\|\widehat{x_{i,j}} - x_{i,j}\|^2$) augmented with the cross entropy for the presence indicator $y_{i,j}$ (CE: $y_{i,j} \log \widehat{y_{i,j}} + (1-y_{i,j}) \log (1-\widehat{y_{i,j}})$). The training strategy comes into three modes. In independent mode, each neural net receives as input the (ground truth) distribution $D_i$ and computes the loss associated with the predicted $\widehat{D_{i+1}}$, noted $\mathcal{L}(\widehat{D_{i,i+1}}, D_{i+1})$. The weakness of the independent mode is that it does not account for the propagation of the errors along the successive segments.
In end-to-end mode (E2E), each neural net receives as input the output of the previous segment, and the associated loss is noted $\mathcal{L}(\widehat{D_{0,i+1}}, D_{i+1})$.
In hybrid mode, the overall loss is a weighted combination of the independent and E2E losses:

$$\mathcal{L} = \sum_{i=0}^{N-1} \lambda \mathcal{L}(\widehat{D_{i,i+1}}, D_{i+1}) + (1-\lambda)\mathcal{L}(\widehat{D_{0,i+1}}, D_{i+1}) \tag{1}$$

**Experimental setting.** The goal of the experiments is to comparatively investigate the merits of the different architectures and losses, and the impact of the trade-off weight $\lambda$ on the overall performance. The overall dataset includes 14000 simulations, computed by Astra [6] and uniformly divided into a training set (80%), a validation set (10%) and a test set (10%). The overall dimension of the control settings is 24 [11].

The LINACNET model is trained on an Nvidia A100 SXM4 80 Go GPU. [2] The impact of the E2E *vs* independent loss is investigated by varying the trade-off parameter $\lambda$ in $[0, 1]$. The inference time per

[2] Using the RAdam optimizer [10], with dynamic learning rate adjustments (halving every ten epochs upon stagnation of validation loss), with batch size 16, aggregated gradients across 4 batches and gradient clipping at 10 for stability. Training is stopped after 80 hours or after 20 epochs without a loss improvement.

batch is 0.8 seconds, significantly faster than the 192 minutes required for a 1-core CPU to simulate the same 16 instances in the batch with Astra.

## 4 Results

**Hybrid loss, Overall results.** For a hybrid loss with $\lambda = .5$, using architecture LINACNET.(a), the particle predictions at the output of the accelerator are quite good, as measured from the R2 indicator above .99 (Table 1) and as visually evidenced in Fig. 2. Most surprisingly, however, it turns out that the end predictions are slightly (but statistically significantly) better than the predictions at the end of the first segment (bottom line in Table 1). A tentative interpretation of this fact is that the first segment appears to be the most complex one, particularly from a numerical viewpoint, as it increases the particle speed from 0 to a fraction of the speed of light in this only segment.

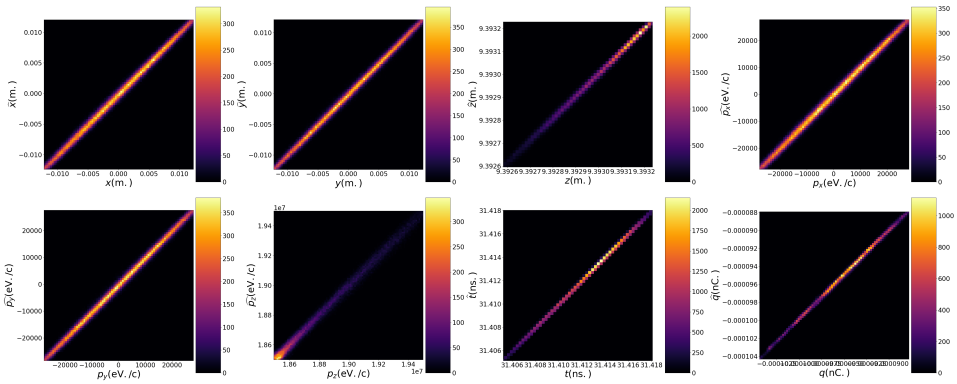



Figure 2: Predicted vs. actual values of the particle coordinates at the end of the accelerator for LINACNET.(a) with hybrid loss ($\lambda = 0.5$). The spread of data points along the diagonal line underscores the low disparity of errors for the particle beam coordinates.

Table 1: LINACNET.(a), hybrid loss ($\lambda = .5$): R2 score of the model prediction over the 8 particle coordinates.

| | $x$ | $y$ | $z$ | $p_x$ | $p_y$ | $p_z$ | $t$ | $q$ |
|---|---|---|---|---|---|---|---|---|
| $\mathrm{R}^2\left(\widehat{D}_{0,N}, D_N\right)$ | 0.9918 | 0.9934 | 0.9999 | 0.9926 | 0.9926 | 0.9989 | 0.9999 | 0.9978 |
| $\mathrm{R}^2\left(\widehat{D}_{0,1}, D_1\right)$ | 0.9847 | 0.9780 | 0.9924 | 0.9754 | 0.9754 | 0.9871 | 0.9933 | 0.9956 |

**Impact of the architecture.** The histogram of the independent losses, aggregated over all segments on the test samples and displayed in Fig. 3, shows that LINACNET.(a) dominates LINACNET.(b), with a difference that is significant at level $10^{-4}$ after paired t-test: the impact of the control settings associated with each segment is slightly better taken into account when provided to the PointNet module. The significant overlap of the two histograms however suggests that the difference is not critical for the overall result.

**Impact of the architecture.** The histogram of the independent losses aggregated over all segments on the test samples and displayed in Fig. 3, shows that LINACNET.(a) dominates LINACNET.(b), with a difference that is significant at level $10^{-4}$ after paired t-test: the impact of the control settings associated with each segment is slightly better taken into account when provided to the PointNet module. The significant overlap of the two histograms suggests that the difference is not critical for the overall result.

The Siamese architecture performs significantly worse than LINACNET on the first segment (Table 2). This result confirms the importance of particle interactions regarding the modeling of an accelerated beam. The GNN-like model performance, though better than the Siamese, is lower than the LINACNETS; this suggests that particle interactions extend beyond local neighborhoods, emphasizing the advantage of modeling particle interactions at the global, PointNet-like level.

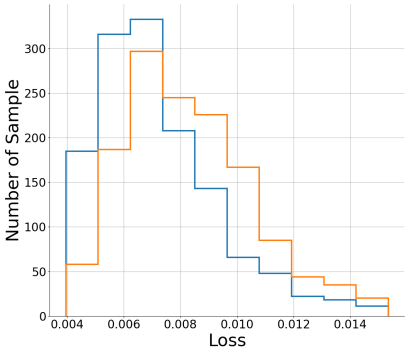



Figure 3: Impact of the LINACNET architecture (LINACNET.(a) in blue, LINACNET.(b) in orange): histograms of independent losses over all segments ($\lambda = 1$).

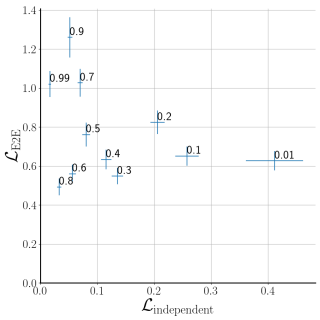



Figure 4: Independent and E2E losses for different values of $\lambda$. The markers' size indicates the error's standard deviation across the test set (1400 samples).

Table 2: Comparative validation of LINACNET and baseline architectures on the first segment.

| Architecture | Loss |
|---|---|
| LINACNET.(a) | $2.43 \cdot 10^{-5}$ |
| LINACNET.(b) | $2.29 \cdot 10^{-5}$ |
| Siamese | $2.62 \cdot 10^{-4}$ |
| GNN-like | $1.08 \cdot 10^{-4}$ |

**Impact of the training strategy.** The impact of the trade-off parameter $\lambda$ is depicted in Fig. 4, with two lessons. Firstly and as expected, the independent loss's magnitude increases when the associated weight $\lambda$ decreases. Secondly, and more surprisingly, the magnitude of the E2E loss cannot be decreased to 0, whatever the value of $\lambda$. Complementary experiments, taking inspiration from [15] and considering the simulator modeling as the multi-objective optimization problem aimed at minimizing both the independent and the E2E losses, have been conducted with a sophisticated adaptive schedule of the trade-off weight $\lambda$, to no avail. Furthermore, complementary experiments with $\lambda = 1$ (independent loss only) show that the E2E loss increases up to $10^5$ because the errors done in each segment cumulate and exponentially increase.

## 5 Discussion and Perspectives

The main contribution of this paper is to demonstrate the feasibility of predicting the whole beam behavior in a particle accelerator. This result, original to our best knowledge, relies on adapting computer vision approaches to the particle physics domain.

The presented experiments and the discussion shed some light on the difficulty of the compound optimization problem associated with modelling the different segments, opening several perspectives for further research. The first perspective is to design a curriculum learning schedule, gradually optimizing the $i$-th neural net using E2E loss while freezing the $i' < i$ neural modules. Another perspective, building upon the fact that a pretty small weight on the E2E loss prevents the explosion of the errors, is to revisit the projection of the gradients proposed in [15]: the goal is to avoid any progress made on the independent losses from deteriorating the E2E loss.

## References

[1] Andreas Adelmann. "On Nonintrusive Uncertainty Quantification and Surrogate Model Construction in Particle Accelerator Modeling". In: *SIAM/ASA Journal on Uncertainty Quantification* 7.2 (Apr. 2019), pp. 383–416.

[2] Renato Bellotti, Romana Boiger, and Andreas Adelmann. “Fast, Efficient and Flexible Particle Accelerator Optimisation Using Densely Connected and Invertible Neural Networks”. In: *Information* 12.9 (2021).

[3] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. “Signature Verification using a "Siamese" Time Delay Neural Network”. In: *Advances in Neural Information Processing Systems*. Ed. by J. Cowan, G. Tesauro, and J. Alspector. Vol. 6. Morgan-Kaufmann, 1993.

[4] Auralee Edelen, Nicole Neveu, Matthias Frey, Yannick Huber, Christopher Mayes, and Andreas Adelmann. “Machine learning for orders of magnitude speedup in multiobjective optimization of particle accelerator systems”. In: *Phys. Rev. Accel. Beams* 23 (4 Apr. 2020), p. 044601.

[5] Auralee Edelen, Nicole Neveu, C Mayes, C Emma, and D Ratner. “Machine learning models for optimization and control of x-ray free electron lasers”. In: *NeurIPS Machine Learning for the Physical Sciences Workshop*. Dec. 2019.

[6] K. Flöttmann. *ASTRA: A space charge tracking algorithm, manual, 2017*. Tech. rep. 2017.

[7] Dan Hendrycks and Kevin Gimpel. *Gaussian Error Linear Units (GELUs)*. 2023. arXiv: 1606.08415 [cs.LG].

[8] Andrei Ivanov and Ilya Agapov. “Physics-based deep neural networks for beam dynamics in charged particle accelerators”. In: *Phys. Rev. Accel. Beams* 23 (7 July 2020), p. 074601.

[9] M. Kranjcevic, B. Riemann, A. Adelmann, and A. Streun. “Multiobjective optimization of the dynamic aperture using surrogate models based on artificial neural networks”. In: *Phys. Rev. Accel. Beams* 24 (1 Jan. 2021), p. 014601.

[10] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. “On the Variance of the Adaptive Learning Rate and Beyond”. In: *International Conference on Learning Representations*. 2020.

[11] H Purwar, E Goutierre, H Guler, M Rossetti Conti, S Chance, A Gonnin, H Monard, A Bacci, M Sebag, J Cohen, and C Bruni. “Random error propagation on electron beam dynamics for a 50 MeV S-band linac”. In: *Journal of Physics Communications* 7.2 (Feb. 2023), p. 025002.

[12] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[13] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017.

[14] A. Scheinker. “Adaptive machine learning for time-varying systems: low dimensional latent space tuning”. In: *Journal of Instrumentation* 16.10 (Oct. 2021), P10008.

[15] Ozan Sener and Vladlen Koltun. “Multi-Task Learning as Multi-Objective Optimization”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc., 2018.

[16] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. “Multi-View Convolutional Neural Networks for 3D Shape Recognition”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015.

[17] Helmut Wiedemann. *Particle accelerator physics*. Springer Nature, 2015.

[18] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. “Point Transformer V2: Grouped Vector Attention and Partition-based Pooling”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh. Vol. 35. Curran Associates, Inc., 2022, pp. 33330–33342.

[19] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. “3D ShapeNets: A Deep Representation for Volumetric Shapes”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015.

[20] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H.S. Torr, and Vladlen Koltun. “Point Transformer”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 16259–16268.