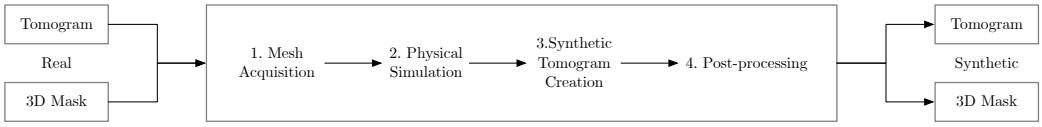
Graphical Abstract

Synthetic Particle Pack Generation for Augmentation and Testing in Geological Tomographic Segmentation

Bogong Wang, Andrew Kingston, Philipp D. Lösel, Warren Creemers



Highlights

Synthetic Particle Pack Generation for Augmentation and Testing in Geological Tomographic Segmentation

Bogong Wang, Andrew Kingston, Philipp D. Lösel, Warren Creemers

- We introduced a novel workflow that generates synthetic tomograms with segmentation ground truths. This method enriches existing tomographic datasets, offering capabilities for dataset augmentation and the evaluation of geological segmentation model.

Synthetic Particle Pack Generation for Augmentation and Testing in Geological Tomographic Segmentation

Bogong Wang[a], Andrew Kingston[b], Philipp D. Lösel[b], Warren Creemers[a]

[a]*School of Computing, The Australian National University, Canberra, 2617, ACT, Australia*
[b]*Research School of Physics, The Australian National University, Canberra, 2617, ACT, Australia*

Abstract

3D imaging of geological particle samples by computed tomography (CT) offers the means for non-destructive analysis. However, obtaining such tomograms with the corresponding segmentation labels remains a significant challenge. This study introduces a novel physics-based simulation workflow that generates synthetic tomograms with segmentation ground truths. The synthetic dataset generation tool produces realistic particle pack tomograms in large quantities, supporting data augmentation and serving as a benchmark for geological tomographic segmentation testing.

Keywords: Computed Tomography, Synthetic Data Generation

1. Introduction

In the field of geology, the use of computed tomography (CT) to scan geological particle samples has become increasingly common, such as reservoir rocks [1], sandstones [2] and ores [3]. CT scanning offers the significant advantage of providing 3D non-destructive inspection of large batches of particle samples, often referred to as particle packs. However, to perform comprehensive analyses of these particle packs, it is essential to accurately segment individual particles. This segmentation is crucial for examining specific attributes such as composition and physical properties. In addition, obtaining such segmentation is often the bottleneck in tomographic analysis pipeline.

With the growing utilisation of machine-learning based segmentation algorithms, the demand for comprehensive datasets has significantly increased. A key challenge in this context is the acquisition of CT volumes along with

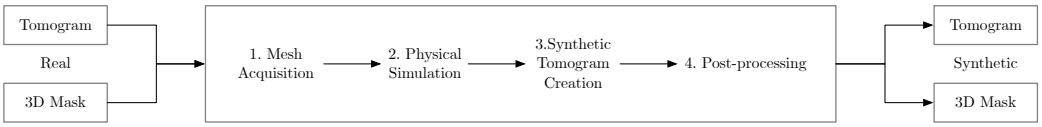


Figure 1: Particle Pack Synthesis Workflow

their precise segmentation ground truths. However, obtaining tomograms requires complex processes, highly skilled technicians, and sophisticated equipment. Moreover, research groups focused on developing segmentation algorithms, may lack access to such CT datasets.

Synthetic data has been extensively utilised for decades to enhance existing datasets and evaluate the performance of computer vision algorithms, including segmentation [4]. This involves generating data that closely resemble real samples in structure, distribution, and essential characteristics, but are computationally produced. Simulation-based methods are widely employed for this purpose [5], leveraging computational models such as physics simulators or scene rendering engines to create synthetic data tailored to specific domains. These methods aim to replicate real-world scenarios or physical phenomena by constructing virtual environments that mimic real-world dynamics with high fidelity. Nevertheless, there is a lack of research on synthesising particle pack tomograms and their corresponding segmentations.

In this work, we present a novel physics-simulation based workflow for generating 3D particle pack tomograms along with their corresponding segmentation ground truths. This approach provides a effective tool for augmenting tomographic segmentation datasets to enhance the training of machine learning-based segmentation models. Furthermore, the synthesised dataset serves as a reliable benchmark for evaluating and validating segmentation algorithms.

In this paper, we begin by introducing the workflow developed for synthesising particle pack datasets. Next, we present experiments demonstrating how these datasets can augment segmentation model training, enhancing model performance. Finally, we discuss the implications of our findings, highlighting their potential to improve segmentation accuracy and streamline processes in relevant applications.

2. Method

To illustrate our proposed workflow for generating synthetic particle packs (depicted in Figure 1), we designed a process that closely replicates the physical steps involved in preparing particles for CT scanning. This workflow encompasses four key stages:

1. Mesh Acquisition
2. Physical Simulation
3. Synthetic Tomogram Creation
4. Post-Processing

2.1. Mesh Acquisition

Mesh acquisition involves extracting each particle's volume from a tomogram and converting it into a mesh for rigid body simulations. As illustrated in Figure 2, the process starts with a particle pack tomogram and its mask. Using the mask, we isolate each particle's volume and label volume, then convert these into mesh format with marching cubes algorithm [6]. To optimise for simulations, particle meshes can be simplified using mesh decimation techniques [7]. This reduces vertex counts while retaining essential geometry and improving simulation efficiency.

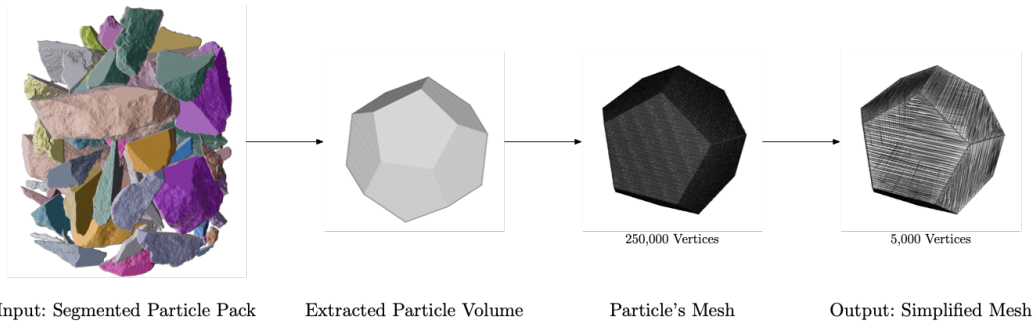


Figure 2: Mesh Acquisition Workflow

2.2. Physical Simulation

The physical particle packing process begins with sieving to separate particles by size. Selected particles are then placed into a container to form a particle pack, ready for CT scanning. Our goal in the physical simulation is

to replicate this process. This is achieved by inputting particle meshes with specific size then using simulator to mimic the particle placement process, i.e., pouring particles into a container. Lastly obtain the placement (location and rotation) of each particle in the simulated particle pack.

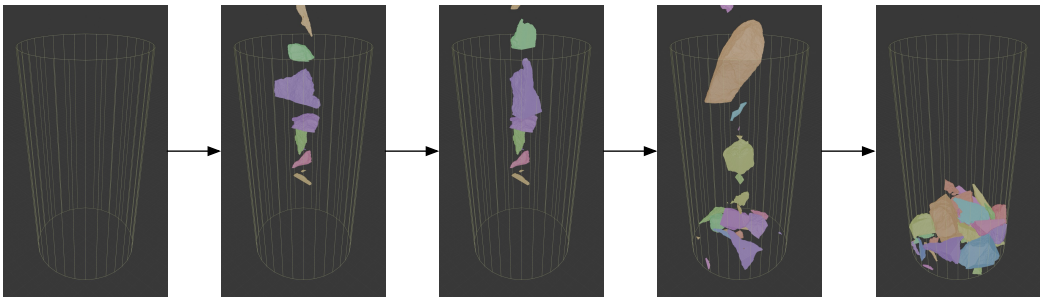


Figure 3: Pipeline of Batched Rigid Body Simulation

The batched rigid body simulation pipeline includes four steps (illustrated in Figure 3):

1. Create Container: Construct a tube-like container, similar to the one used in CT scans.
2. Add Particle Meshes: Introduce particle meshes into the simulation environment.
3. Set Meshes' Properties: Assign each particle mesh properties such as weight, initial position, and scale (details in **??**).
4. Rigid Body Simulation: Start the simulation and allow the particle meshes to settle after a defined number of timesteps. The simulator will then output the final results, which contains each mesh's translation and rotation. Rigid body simulation is used to model this process because it approximates the motion and interaction of particles under realistic conditions, capturing essential dynamics such as collision, gravity, and friction without deforming the individual meshes.

After evaluating multiple options, including Unity 3D [8], Unreal Engine [9], and PhysX [10], we chose Blender [11] for conducting batched rigid body simulations. Blender stood out due to its accessible graphical user interface (GUI), which facilitated the visual inspection of packed particle formations.

This helped us verify whether the simulation accurately represents the desired physical characteristics. Additionally, Blender's scripting interface allowed seamless integration into a Python-driven pipeline, ensuring that the project maintained a consistent and adaptable programming environment.

In simulation of the particle packing processes, specific settings and procedures are crucial to achieving accurate and reproducible results. The implementation specifics used in our simulations include:

- Particle weights affects collision between particles and container in our particle packing simulation. In our implementation, we assume each particle has a uniformity and even quality. Then the mass of each particle is estimated based on particles' density and the size of the mask. This method provides a rapid approximation of weight, suitable for scenarios where detailed precision is less critical.

- The initial positioning and scaling of particle meshes are highly dependent on the experimental needs. These adjustments significantly affect the final placement of particles within the volume.

2.3. Synthetic Tomogram Creation

Based on the simulation results and particle volumes, the synthetic particle pack can be created. We have a list of particle positions and orientations (rotations) from step 3.2. We also have the volumetric data of each partcile from step 3.1. Given this, the synthetic tomogram creation is straightforward and involves three steps:

1. Creating the target simulation container
2. Rotating each particle volume according to the simulation results
3. Placing each rotated particle volume in the simulation container based on the physics simulation results form 3.2

2.4. Post-processing

Post-processing plays a crucial role in bridging the gap between synthetic and real-world data, enhancing the realism of synthetic tomograms. In our implementation, Gaussian noise is commonly added to replicate real-world imperfections, thereby producing a more realistic synthetic dataset. Artefacts such as motion blur, ring effects, or beam hardening, which are common in tomographic imaging. By simulating these artefacts, synthetic datasets can closely mimic the challenges faced when processing real-world

data, providing an invaluable ground truth for developing algorithms that are robust to these artefacts.

An example of the post-processed synthesised slice is shown in Figure 4. After post-processing, the synthetic slice appears more realistic.

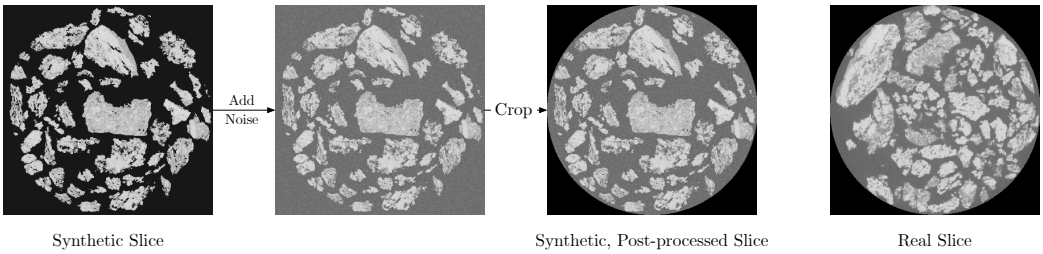


Figure 4: An Example of Postprocessing Slice in Simulated Particle Pack Data

3. Experiment & Discussions

Having constructed synthetic particle packs as described in Section 2. We will conduct a couple of experiments to investigate the efficacy of using synthetic particle packs to augment existing training datasets for 2D particle segmentation. This will be exemplified by comparing the performance gap between models trained on raw and augmented datasets.

3.1. Experiment Setup

To effectively train and evaluate our model, we partitioned each tomogram into training and testing sets. We specifically avoided random selection of slices for the testing set due to the high similarity between consecutive slices within the particle pack. This similarity can lead to overestimation of model performance, as the model might simply be recognising similar instances it has already encountered in adjacent slices, rather than generalising from diverse examples. Therefore, we allocated the central 20% of the particle pack slices to the testing set, and the remaining 80% for training. This will avoid the similarity between training set and testing set.

In this experiment, the YOLOv8m-seg model from Ultralytics [12] will be employed due to its versatility across a wide range of applications as well as optimal balance between computational speed and segmentation accuracy.

Lastly, we selected the Dice score [13], also known as Dice-Sørensen coefficient (shown in eq. (1)) to evaluate segmentation performance of trained models. As it effectively evaluating segmentation accuracy by accounting for

	Post-processing	Real Large	Real Small
Real Large	n/a	94.5%	77.0%
Real Large + Syn	✗	94.9%	79.6%
Real Large + Syn	✓	96.0%	80.1%
Real Small	n/a	92.4%	90.1%
Real Small + Syn	✗	92.4%	88.8%
Real Small + Syn	✓	94.8%	90.9%

Table 1: Segmentation performance between models trained on different datasets. Each row represents the training set, and each column represents the testing set (except for the "Post-processing"" column). The "Post-processing" column indicates whether the augmented dataset has been post-processed. Real Large refers to a real tomogram dataset with large particles, while Real Small refers to a real tomogram dataset with small particles. Rows include results from training on only the real datasets and combinations with synthetic data (Real Large + Syn, Real Small + Syn). Columns indicate testing on real datasets with large or small particles.

both false positives and false negatives, providing a balanced view of under and over-segmentation.

$$\text{Dice Score} = 2 \times \frac{\text{Area of Overlap}}{\text{Total Area}} \tag{1}$$

3.2. Result I: Postprocessing Is Crucial In Augmentation

In our experiment, the post-processing involved adding Gaussian noise to the synthetic data. The observed results from Table 1 indicate that after post-processing, the training dataset was successfully augmented, leading to improvements in the models' performance across various test sets. Conversely, without adding noise, the models' performance sometimes degraded (from 90.1% to 88.8%).

This can be attributed to the noise making the synthetic slices more realistic and closer to the actual data the model will encounter, thereby reducing the domain gap between synthetic and real data. Without noise, the synthetic data lacks the necessary realism, leading the model to learn features that do not transfer well to real-world scenarios. This discrepancy can cause the model to perform poorly on real data, effectively learning in a different domain than the one it is tested on. This highlights the importance of appropriate postprocessing to make synthetic datasets more realistic, which enhances its utility for augmenting existing datasets.

3.3. Result II: Synthetic Data Improves Segmentation Accuracy

To assess performance improvements with synthetic particle packs, we trained and tested segmentation models using real data alone, and real data augmented with synthetic particle pack data. The results for each model, presented in Table 1, demonstrate that incorporating synthetic data improves segmentation performance under most scenarios. Specifically, when trained with real large particles augmented with synthetic data, the accuracy increased from 94.5% to 96.0% for large particles and from 92.4% to 94.8% for small particles. Similarly, training with real small particles combined with synthetic data improved the accuracy on large particles from 77.0% to 80.1%, while the improvement for small particles was more modest, rising from 90.1% to 90.9%. These results indicate that synthetic data enriches the training set with diverse examples, thereby enhances the model's ability to segment tomograms with different particle sizes.

4. Conclusion

This paper presents a comprehensive workflow for generating synthetic particle pack datasets through physics-based simulation. Our approach addresses the challenges of acquiring accurate particle pack tomograms and segmentation ground truths in geology. These tomogram-mask pairs can be used to augment existing segmentation datasets and evaluating segmentation algorithms. The integration of noise and artefacts in post-processing further bridges the gap between synthetic and real-world data, offering a data generation platform for future research.

Our experimental results validate the effectiveness of our synthetic particle pack generation workflow. By augmenting real datasets with synthetic data, we achieved consistent improvements in segmentation accuracy across different particle sizes, as demonstrated by higher Dice scores in Table 1. The integration of post-processing techniques, specifically the addition of noise, was crucial in bridging the gap between synthetic and real-world data, leading to enhanced model performance (Table 1). These findings highlight the potential of our approach to enrich annotated particle pack data for training and validation of segmentation algorithms.

In the current synthetic particle-pack-generation workflow, we rely on manually labelled particle packs, which limits the versatility of the particle data. To eliminate this dependency and increase data diversity, we propose two improvement directions. The first approach involves using classical

methods such as free-form deformation (FFD) [14] to alter particles' shape while retaining their core characteristics. This method is relatively fast, easy, and deterministic but still relies on the existing particle data. The second approach involves employing advanced machine-learning techniques, such as diffusion models [15] or generative adversarial networks (GANs) [16]. These models would first learn the features of the particle data and then generate entirely synthetic particle data.

References

[1] M. Van Geet, R. Swennen, M. Wevers, Quantitative analysis of reservoir rocks by microfocus x-ray computerised tomography, Sedimentary Geology 132 (1) (2000) 25–36. doi:https://doi.org/10.1016/S0037-0738(99)00127-X.

[2] V. Cnudde, J. Dewanckele, W. De Boever, L. Brabant, T. De Kock, 3d characterization of grain size distributions in sandstone by means of x-ray computed tomography, in: Quantitative mineralogy and microanalysis of sediments and sedimentary rocks, Vol. 42, Mineralogical Association of Canada (MAC), 2012, pp. 99–113.

[3] M. Warlo, G. Bark, C. Wanhainen, A. Butcher, F. Forsberg, H. Lycksam, J. Kuva, Multi-scale x-ray computed tomography analysis to aid automated mineralogy in ore geology research, Frontiers in Earth Science 9 (2021) 789372. doi:10.3389/feart.2021.789372.

[4] S. R. Richter, V. Vineet, S. Roth, V. Koltun, Playing for data: Ground truth from computer games (2016). arXiv:1608.02192.
URL `https://arxiv.org/abs/1608.02192`

[5] C. M. De Melo, A. Torralba, L. Guibas, J. DiCarlo, R. Chellappa, J. Hodgins, Next-generation deep learning based on simulators and synthetic data, Trends in Cognitive Sciences 26 (2) (2022) 174–187. doi:10.1016/j.tics.2021.11.008.

[6] W. E. Lorensen, H. E. Cline, Marching cubes: A high resolution 3d surface construction algorithm, SIGGRAPH Comput. Graph. 21 (4) (1987) 163–169. doi:10.1145/37402.37422.
URL `https://doi.org/10.1145/37402.37422`

[7] M. Garland, P. S. Heckbert, Surface simplification using quadric error metrics, in: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., USA, 1997, pp. 209–216. doi:10.1145/258734.258849.
URL https://doi.org/10.1145/258734.258849

[8] Unity Technologies, Unity Game Engine, https://unity.com (2024).

[9] Epic Games, Unreal Engine, https://www.unrealengine.com (2024).

[10] NVIDIA Corporation, NVIDIA PhysX, https://developer.nvidia.com/physx-sdk (2024).

[11] Blender Foundation, Blender Python API: Object Operators, https://docs.blender.org/api/current/bpy.ops.object.html#bpy.ops.object.origin (2023).

[12] G. Jocher, A. Chaurasia, J. Qiu, Ultralytics YOLO (Jan. 2023).
URL https://github.com/ultralytics/ultralytics

[13] L. R. Dice, Measures of the amount of ecologic association between species, Ecology 26 (1945) 297–302.
URL https://api.semanticscholar.org/CorpusID:53335638

[14] T. W. Sederberg, S. R. Parry, Free-form deformation of solid geometric models, in: Proceedings of the 13th annual conference on Computer graphics and interactive techniques, 1986, pp. 151–160.

[15] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models (2020). arXiv:2006.11239.
URL https://arxiv.org/abs/2006.11239

[16] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative Adversarial Networks (Jun. 2014). arXiv:1406.2661.