Title
Numerical Modeling of Soil Fabric of Naturally Deposited Sand

Author
Tan, Peng

Publication Date
2022

Peer reviewed|Thesis/dissertation

Numerical Modeling of Soil Fabric of Naturally Deposited Sand

by

Peng Tan

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Civil and Environmental Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Nicholas Sitar, Chair
Professor Kenichi Soga
Professor Tarek Zohdi

Spring 2022

Numerical Modeling of Soil Fabric of Naturally Deposited Sand

Abstract

Numerical Modeling of Soil Fabric of Naturally Deposited Sand

by

Peng Tan

Doctor of Philosophy in Engineering - Civil and Environmental Engineering

University of California, Berkeley

Professor Nicholas Sitar, Chair

This dissertation provides a systematical investigation of computational approaches to modeling of granular materials. Granular materials are ubiquitous in everyday life and in a variety of engineering and industrial applications. Despite the apparent simplicity of the laws governing particle scale interactions, predicting the continuum mechanical response of granular materials still poses extraordinary challenges. This is largely due to the complex history dependence resulting from continuous rearrangement of the microstructure of granular material, as well as the mechanical interlocking due to grain morphology and surface roughness. X-Ray Computed Tomography (XRCT) is used to characterize the grain morphology and the fabric of the granular media, naturally deposited sand in this study. The Level-Set based Discrete Element Method (LS-DEM) is then used to bridge the granular behavior gap between the micro and macro scale. The LS-DEM establishes a one-to-one correspondence between granular objects and numerical avatars and captures the details of grain morphology and surface roughness. However, the high fidelity representation significantly increases the demands on computational resources. Herein, we introduce an enhanced image processing workflow for XRCT images in order to optimize the grain and fabric resolution. A parallel version of LS-DEM is then introduced to significantly decrease the computational demands. The code employs a binning algorithm, which reduced the search complexity of contact detection from $O(n^2)$ to $O(n)$, and a do- main decomposition strategy is used to elicit parallel computing in a memory- and communication- efficient manner. The parallel implementation shows good scalability and efficiency.

High fidelity LS avatars obtained from XRCT images of naturally deposited sand are then used to replicate the results of triaxial tests using the new parallel LS-DEM. Both micro- and macro-mechanical behaviors of natural materials were well captured and validated with experimental data. The results of the numerical modeling show that the primary source of peak strength of sand is the mechanical interlocking between irregularly shaped grains. Flexible membrane simulations with a rotatable loading platen were found to accurately

match experimentally observed relationships between deviatoric stress and mobilized friction angle with axial shortening for naturally deposited sand. Finally, we investigated the viability of modeling dynamic problems with newly formulated impulse-based LS-DEM. The new formulation is stable, fast and energy conservative, however, it may be numerically stiff when the assembly has a substantial mass difference or badly reconstructed particles as a result of poor image resolution. We also demonstrated the feasibility of modeling deformable structures in the rigid body framework and proposed several enhancements to improve the convergence of collision resolution, including a hybrid time integration scheme to separately handle at rest contact and dynamic collision. We also extended the impulse-based LS-DEM to include arbitrarily shaped topography surfaces and exploited algorithmic advantages to investigate interactions between topography and colliding objects. The novel formulation significantly improves performance and allows for larger timesteps, which is advantageous for observing the full development of physical phenomena such as rock avalanches.

To mom, dad and sister
给妈妈，爸爸和姐姐

Mom, I promised you I will get a PhD. Family is not an important thing. It's everything.
妈，我答应你要拿一个博士的

Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank my advisor, Prof. Nicholas Sitar, for his persistent guidance and encouragement as I pursued scientific curiosity freely. I am grateful that Prof. Sitar accepted me into his group and taught me everything, beginning with computer assembly and English practice. He has offered a wealth of advice and some of which I am certain to carry with me for the rest of my life. He also educated me how to think independently, communicate efficiently and plan strategically. Feeling confused and frightened, I spent half a year in a shipping container chamber at GCF, HKUST, contemplating what I genuinely want to do in my twenties. Pursuing a PhD was the best decision I have ever made, and I have treasured memories with his group in this lovely place.

I greatly appreciate Prof. Kenichi Soga, Prof. Tarek Zodhi, Prof. James Demmel and Prof. Jon Wilkening for their willingness to serve on my qualifying/thesis committee. I know Prof. Soga before I arrived in Berkeley, and I have benefited greatly from his help and valuable advice. I have been benefited greatly from Prof. Zohdi's valuable advice on simulating particulate flow and deformable network. I have particularly enjoyed Prof. Demmel and Prof. Wilkening's class, which piqued my long-standing interest in mathematics and the knowledge I learned from them forms the foundation for substantial portion of my research.

I believe I learned a lot from with Prof. Jonathan Bray, Prof. Robert Kayen, Prof. Lin Lin, Prof. Sanjay Govindjee in my first year. Their classes enable me to segue smoothly from employment to research.

I would want to express my gratitude to Prof. José Andrade and Dr. Reid Kawamoto for granting me access to the LS-DEM code and patiently addressing my inquiries. I also owe gratitude to Prof. Michael Gardner and Prof. Fernando Garcia for their constructive suggestions in the early stages of my PhD research.

I would like to acknowledge the National Science Foundation under grant CMMI-1853056, the Edward G. and John R. Cahill Chair, the Berkeley-France Fund, and PEER-Caltrans at UC Berkeley for funding my PhD.

I would like to thank Prof. Jidong Zhao, Prof. Chao Zhou, and Prof. Andy Leung for serving as guarantors for my graduate studies, which ignites my dream and propels me to heights I never imagined.

I convey my heartfelt appreciation to everyone with whom I had the privilege of working and living in Berkeley: Zhenxiang SU, Xinyi Qian, Jinyan Zhao, Bodhinanda Chandra, Hasitha Sithadara Wijesuriya, and Yuval Keissar.

I am grateful for those who assisted me in Hong Kong: Huan Wang, Qijie Ma, Chao Wang, Junnan Zhang, and Haowen Guo. I will never forget the moments we had and how they formed me as a person.

Special thanks to Yu Du, Jiazhi Kang, Congyu Yu, Yan Zhang, Xiaocheng Wei and Hanlin Guan; our Wechat group is always a safe haven for me. They aided me at various points and made my life easier.

Additionally, the road trip to Seattle with Yue Sun and Haolin Fu is a memorable aspect of my experience, aside from my strenuous study, and I am looking forward to embarking on another journey.

Lastly, I would like to dedicate this work to my sister, mom, and dad. Words cannot do justice to the amount of love and support that I have received during the last 27 years. They have never once questioned my decisions and choices; this accomplishment would not have been possible without their support and sacrifices.

Chapter 1

Introduction

1.1 Motivation and Background

The purpose of this dissertation is to present an investigation into the char- acterization and simulation of the fabric of naturally deposited sands at the micro scale. The importance and influence of the geologic setting and of the depositional environment on the mechanical properties of granular deposits, sands, gravels, and silts has been long recognized in geotechnical engineering (Terzaghi, 1955). Since then, numerous investigators have examined the influence of the depositional fabric on the mechanical properties of sands prepared in the laboratory and showed that sand samples prepared by different methods exhibited different mechanical response even when the sand was compacted to the same relative density (See e.g. Oda, 1972; Mulilis et al., 1977; Ochiai & Lade, 1983; Lam & Tatsuoka, 1988; Ishihara, 1993; Zlatovic & Ishihara, 1997; Yoshimine et al., 1998; Vaid et al., 1999; Wood & Maeda, 2008). Most importantly, all of these studies involved samples reconstituted in the laboratory. Thus, the focus of this study was naturally deposited sand.

The opportunity for the study of the fabric and of the mechanical properties of naturally deposited sand arose as a result of a geotechnical investigation of the liquefaction potential of the hydraulic fill on Treasure Island in the San Francisco Bay. The island was constructed by hydraulic filling over a natural, sand shoal deposit in the San Francisco Bay in the late 1930's. The investigation revealed that the natural, shoal sand had much higher liquefaction resistance than the hydraulic fill, pointing to the difference in the depositional fabric of the two deposits as the most likely explanation for the observed difference in liquefaction resistance. A scanning electron microscope (SEM) photograph of an undisturbed sample of the shoal sand (Figure 1.1) shows that the sand is tightly packed, and the sand grains are arranged in an interlocking fabric. Garcia et al. (2022) used X-Ray Computed Tomography (XRCT) to obtain detailed images of the grain morphology and of the depositional fabric of the shoal sand and performed a series of small-scale triaxial tests to characterize the mechanical properties of the sand. The results of the triaxial tests (Figure 1.2) show that the peak mobilized friction angle of the undisturbed sand, with its intact depositional fabric,

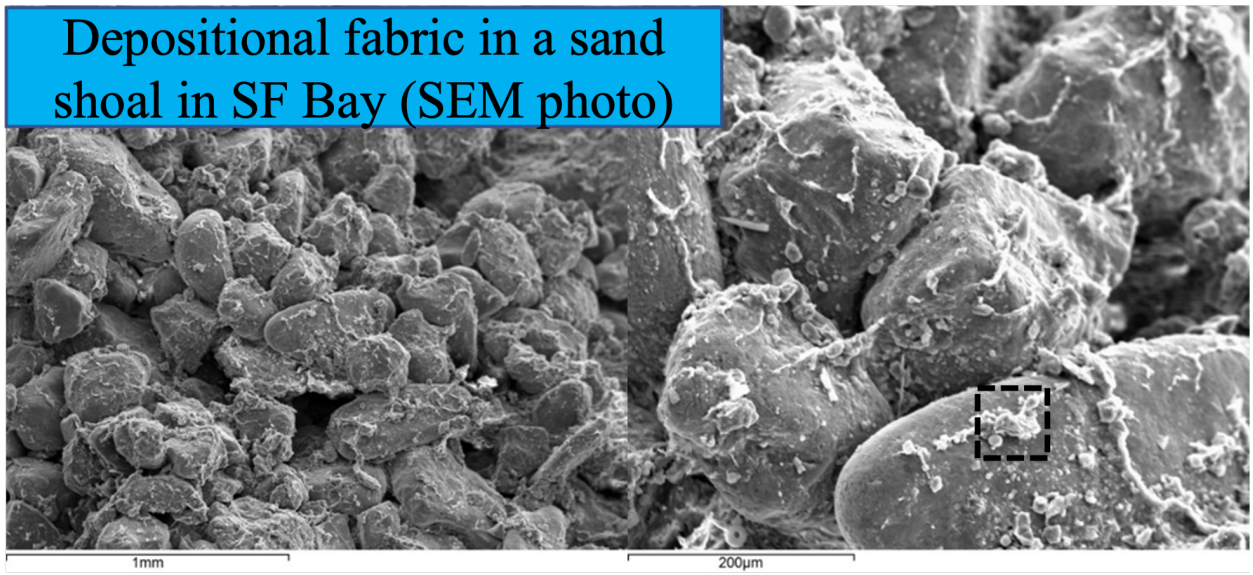

Figure 1.1: SEM image of the depositional fabric in a sand shoal in San Francisco Bay, California (courtesy ENGEO). The black box shows an example of small particles or clay attached on the surface of a large particle.

is significantly higher than that of the disturbed sample, and it also exhibits a much higher initial stiffness.

This new data set consisting of XRCT images and triaxial test results provided the opportunity to model the micromechanical behavior of the naturally deposited material and to explore the available methodology. Specifically, Kawamoto et al. (2016) transformed an XRCT image of a granular object into a mathematical descriptor (an "avatar") of the grain using a Distance Regularized Level-Set Evolution (DRLSE) formulation proposed by Li et al. (2005) and developed a Level-Set (LS) Discrete Element Method (DEM, Cundall & Strack, 1979) code, LS-DEM. They used LS-DEM to demonstrate the significance of accurately modeling sand grain morphology and modeling its micromechanical behavior in a simulated triaxial test. The high fidelity of the reconstructed sand samples and the ability to faithfully model the micromechanical properties of the sand made it an attractive candidate for the work presented here. The challenge, however, was the large computational cost of the code developed by Kawamoto et al. (2018) which made it prohibitively expensive for analysis of data sets with large numbers of particles. Thus, the objectives of this study were to produce level set avatars from the XRCT scans, develop a parallel LS-DEM code capable of modeling large data sets, and to use the newly developed code to model the mechanical response of the naturally deposited shoal sand.

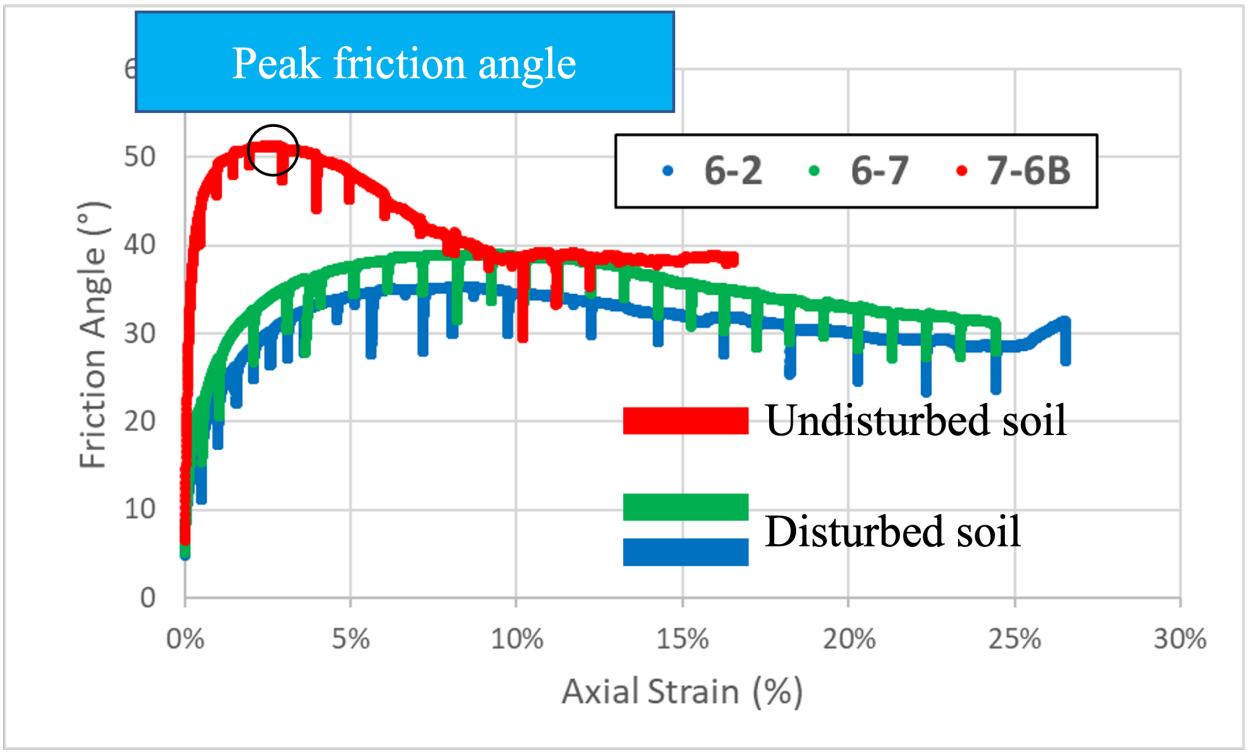

Figure 1.2: Mobilized friction angle and stiffness in triaxial tests on undisturbed and disturbed samples of shoal sand from the San Francisco Bay (from Garcia et al., 2022).

1.2 Thesis Content

The following chapters of this thesis are drafts of papers to be submitted for publication and therefore, some content may be repeated for the sake of completeness.

Chapter 2 is a discussion of image preprocessing workflow that incorporates image denoising, single image super resolution, image segmentation and level-set reconstruction. To begin, we shows that the Non-Local Mean (NLM) filter enhances the visible structures in XRCT images of granular material. We then discuss different techniques for image super-resolution based on sparse signal representation and capable of recovering high-resolution images from low-resolution input images. Finally, an image binarization algorithm based on Hidden Markov Random Fields (HMRF) is presented. This is a statistical model for binarizing particle morphologies from XRCT images that incorporates spatial information and strikes a balance between simple thresholding and more expensive neural networks.

Chapter 3 focuses on the development of a LS-DEM code using a variant of the binning algorithm and a spatial domain decomposition strategy to model arbitrarily complex-shaped grains with a history-dependent contact model. The algorithm is optimized for the existing LS-DEM code, and the majority of the parallel algorithms and implementation details can

also be ported to other disciplines. In this implementation, performance-critical implementation details are optimized for high performance and scalability, and the performance of parallelized LS-DEM is benchmarked in terms of speedup, efficiency, scalability, and granularity for problems of various sizes and using different amount computing resources.

Chapter 4 presents numerical triaxial compression simulations of uncemented sands reconstructed from XRCT images with different resolutions. A calibrated linear elastic model with the Coulomb friction yield criterion is used to simulate the frictional response. A significant conclusion reached in this study is that the primary source of mobilized strength is the result of mechanical interlocking between irregularly shaped grains; thus, even the simplest contact model is consistent with the micro- and macromechanical responses observed in experimental data, provided the microstructure of the grains and grain-to-grain contacts is accurately captured. Flexible membrane simulations with a rotatable loading platen are found to be more accurate at predicting the stress-strain and volumetric response, and the onset and growth of strain localization. Our simulation results match experimentally observed relationships between deviatoric stress and mobilized friction angle as axial shortening increases.

Chapter 5 presents a parallel LS-DEM algorithm and code reformulated in an impulse-velocity framework. The results of numerical experiment are presented to show that the impulse-based method is compelling in large part because it circumvents the need for small time step and saves significant amounts of computational effort for complex shaped objects. To improve convergence of collision resolution, we modify an existing collision resolution algorithm and propose a novel time integration scheme. Additionally, we demonstrate the modeling of protection nets on arbitrarily shaped topography, showing that impulse-based methods can also be used to simulate the dynamics of deformable structures. The results show that the primary limitation of the impulse-based method remains its inability to model a system of irregular, non-convex, non-uniform objects, particularly in a highly confined quasi-static environment where objects of very different shapes and sizes interact at numerous contact points and travel at low speeds.

Chapter 6 is a summary of the major findings, including recommendation for future extensions of the numerical modeling techniques presented herein.

Chapter 2

XRCT Image Processing for Sand Fabric Reconstruction

2.1 Introduction

It is well established that the interaction of individual particles contributes significantly to the macroscopic behavior of granular materials under various loading conditions. In naturally deposited granular materials, sands, silts and clays, the process of deposition in different depositional environments results in a unique arrangement of the particles/grains, referred to as fabric. A good understanding the soil fabric and the grain micromorphology, which includes the surface texture and irregularities, is important in understanding the macroscopic mechanical properties, such as shear strength, dilatancy, crushability, and localization. The SEM image (see Figure 1.1) of a sand sample taken from a sand shoal in the San Francisco Bay illustrates the intimate arrangement of the grains in this type of deposit. The major challenge of imaging and reconstructing this fabric is the small size of the grains on the order of 0.1mm and their close packing which makes it difficult to distinguish the grain boundaries.

In numerical modeling of granular media, Discrete Element Method (DEM, Cundall & Strack, 1979) has been the go-to technique. While spherical grains are readily modeled in DEM, more complex shapes of grains and their packing in natural sands, required creation of more complex objects such as clumped spheres (Garcia et al., 2009; Tamadondar et al., 2019; Wu et al., 2021), and simplex or polyhedron (Zhao et al., 2006; Zhao & Zhao, 2021; Wang et al., 2021). Most recenttly, the level-set (LS, Vlahinić et al., 2014; Kawamoto et al., 2018) approach offers high-fidelity depiction of arbitrary shaped grains, allowing for unparalleled capture of granular materials' macroscopic and local behaviors. LS is preferred over other approaches for describing and reconstructing grain morphology because other widely used but conceptually distinct approaches have inherent disadvantages. For instance, mathematical Fourier descriptors may lose some local information regarding the surface morphology due to the artificial selection of cross-sections, particularly for extremely irregular grains. Whereas the spherical harmonic function is incapable of addressing non-convexity in natural sand

caused by weathering and erosion.

However, the quality of the LS reconstruction is highly dependent on the accuracy with which the true shape of the grains can be captured. The XRCT technique, developed over the last two decades is an excellent tool for 3D imaging and characterization of the microstructure and micromorphology of real sand particles and also for capturing the fabric. While the current workflow has been successful in obtaining the shapes of clean, pluviated sand (Vlahinić et al., 2013; Kawamoto et al., 2018) and clean natural sands (Fu et al., 2017), the image reconstruction becomes a lot more challenging in resolving the individual grains and fabric of naturally deposited fine sands, such as shown in Figure 1.1. Specifically, in this case, the presence of clay adhesions and very fine particles which are clumped on the larger grain surfaces alters the natural shape boundary captured during the XRCT. Moreover, due to the polymineralic nature of the natural sands, the presence of minerals (i.e., Magnetite) opaque to X-Rays makes the reconstruction process even harder and makes it a challenging task to identify the true grain boundaries.

Herein we explore different techniques for processing XRCT images with the aim of improving the grain segmentation process for naturally deposited sands. Firstly, the images were denoised by non-local means filter, this filter identifies and averages neighborhoods across an entire image based on the degree of similarity with the target rather than the proximity. Resulted denoised images preserve edge sharpness and removes noise to a greater degree. Secondly, a learning-based image super-resolution method (Yang et al., 2010) was utilized to enhance the resolution of images based on sparse signal representation. The aim of this procedure is to reduce the partial volume effect due to the low resolution of images. Research on image statistics suggests that image patches can be well-represented as a sparse linear combination of elements from an appropriately chosen over-complete dictionary. This method seeks a sparse coefficient for each patch of the low-resolution input and then uses the coefficient to generate the high-resolution output. Results from the proposed method outperforms the conventional methods of image super-resolution. Lastly, an image segmentation method adopted from Zhang et al. (2001) was proposed to differentiate individual object from background. This method clusters pixel via Hidden Markov Random Field (HMRF) with Weighted Expectation Maximization (WEM) algorithm where each group of pixels (void, water, solid etc.) is modelled as a distinct Gaussian and the spatial connectivity is imposed via a stochastic Markov network. Conventional methods such as Otsu's method (Otsu, 1979), K-means (MacQueen et al., 1967), or histogram analysis can be utilized as prior for the optimal label configuration of an image. Such configuration maximizes the maximum a posterior (MAP) estimation or equivalently minimizes the free energy. In the end, our proposed workflow increases the accuracy and robustness of the image reconstruction, and it leads to much improved binary segmentation of XRCT images.

2.2 Image Denoising

X-Ray images naturally contain noise due to the statistical nature from the generation of photons. Dealing with the noise in the images is the one of the first steps in any image processing workflow. A good denoising algorithm should be able to clean up the local intensity variations while preserving the image sharpness in the areas of importance, e.g., near phase edges or grain-to-grain contacts, perhaps without a prior knowledge of what a noise free image looks like. The fundamental idea behind all noise reduction algorithms is to reduce the statistical noise through an averaging process. Most widely used technique for image denoising use a low-pass filter to decrease the disparity between nearby pixels.

2.2.1 Gaussian Filter

Gaussian filter is one of the widely used linear low pass filter which averages over small window of pixels $I(x+t)$ by weighting the intensity values of the pixels according to discrete Gaussian kernel $G_\rho(t)$ with variance ρ as shown in below. Since Gaussian filter is circularly symmetric it can be treated as two independent one dimensional calculation which reduces the computational complexity from $O(n^2)$ to $O(n)$.

$$I^{GF}(x) = G_\rho(t) * I(x+t) \tag{2.1}$$

$$G_\rho = \frac{1}{C}\exp\left(-\frac{\|x-\mu_x\|^2}{2\rho^2}\right) \tag{2.2}$$

where C is normalization factor, μ_x is the average intensity value of the region of interest.

2.2.2 Median Filter

Another popular filter used in image denoising is median filter which is a nonlinear filter by design. It works by extracting the median value from the neighborhood window of a pixel and assigning it to the corresponding output image pixel. Since this filter does not involve any diffusive actions, median filter is better for removing local outliers. In addition the sharpness of image is better preserved compared to Gaussian filter. Major drawback of both the Gaussian and median filter is, they work by considering only a small local region of the whole image. But in a single XRCT image there are similar regions scattered throughout the image. Therefore, one could take into account all these visually similar regions in the averaging rather than restricting it to the neighbouring pixels, such approach yields a far better result in terms of noise removal and preserving edge sharpness.

2.2.3 Adaptive Filter

Barner et al. (1992) proposed using an adaptive stack filter for denoising, where a series of filters (i.e., median filters and morphological operators like dilation and erosion)

are combined. The stack sequence of filters is determined through experiments on sets of noisy images and their original counterparts. Also, Tomasi and Manduchi (1998) proposed a bilateral filter which accounts for both the geometric and photometric similarity of the neighborhood pixels. The photometric similarity was taken account by assigning smaller coefficients to pixels with high intensity difference and vice versa. Drawback of this method is that it only compares the intensity of a single pixel for comparison making it a less robust method.

2.2.4 Nonlinear Anisotropic Diffusion Filter

Perona and Malik (1990) formulated smoothing as a diffusive process, which is suppressed or stopped at boundaries by selecting locally adaptive diffusion strengths.

$$\frac{\partial}{\partial t} I(x,t) = \nabla \cdot (k(x,t) \nabla I(x,t)) \tag{2.3}$$

The diffusion strength is controlled by $k(x,t)$, where x represents the spatial coordinates in the image denoising context. The variable t is the process ordering parameter, which is the discrete implementation used to enumerate iteration steps. The diffusion function $k(x,t)$ depends on the magnitude of the gradient of the image intensity. It is a monotonically decreasing function which mainly diffuses within regions and does not affect region boundaries at locations of high gradients. One example of such diffusion function is:

$$k(x,t) = \exp\left(-\left(\frac{\|\nabla I(x,t)\|}{K}\right)^2\right) \tag{2.4}$$

This is a monotonic decrease of the diffusion coefficient with increasing gradient, the parameter K is chosen according to the noise level and the edge strength. Gerig et al. (1992) extended NADF to 3D space and produced better noise reduction due to the use of context from larger neighborhood than 2D slices.

2.2.5 Non-Local Mean Filter

Non-Local Means (NLM) filter denoise images by considering the visually similar context over the entire image, this makes it an ideal candidate for our application. A significant improvement over both Gaussian filter and Mean filter can be seen by identifying neighborhoods across an entire image that are of similar context and averaging based on the degree of similarity between these windows, regardless of their proximity to the source. This implies that each single pixel can be expressed as a linear combination of all other pixels. The idea is similar to the bilateral filtering proposed by Tomasi and Manduchi (1998), which accounts both the geometric closeness of the pixels and the photometric similarity between pixels. But NLM filter considers an image patch surrounding the pixel of interest instead of a single

pixel, which makes the method more robust to large noise. The Non-Local Mean Filter can be represented as in the following:

$$I^{NLM}(x) = \frac{1}{C(x)} \int_I w(x,y) * I(y) dy \tag{2.5}$$

$$w(x,y) = \exp\left(-\frac{G_\rho(t) * \|I(x+t) - I(y+t)\|}{2h^2}\right) \tag{2.6}$$

$\|I(x+t) - I(y+t)\|$ measures the Euclidean distance or similarity between source image window $I(x+t)$ with all other windows across the image I, namely $I(y+t)$. It is logical that contextually similar patches are considered closer to each other and therefore assigned higher weight $w(x,y)$. As image features are often presented in terms of local patches, instead of a single pixel value or an entire image, it is more meaningful to calculate the distance between image patches. The resulting similarity $\|I(x+t)-I(y+t)\|$ is further convoluted with a filter, i.e., Gaussian filter or Unit filter if one decides to assign higher weights at the center. For example, less importance is given to pixels away from the center of the window via discrete Gaussian kernel $G_\rho(t)$ and scale the Euclidean distance by a value of noise variance h. Once the weights are collected, perform a weighted average over all window centers $I(y+t)$ and assign the result to the source pixel $I^{NLM}(x)$, i.e., the pixel in the center of the source window. A constant $C(x)$ assures that the sum of all weights $w(x,y)$ is normalized. In this way, averaging is no longer based on proximity or distance from the source pixel, but it is rather a function of the context in which the pixel appears. In practice, searching an entire image to find out all patches with high degree of similarity is computationally expensive, and a trade-off is often made by limiting the size of search window such that the number of similar patches is enough for denoising but still affordable.

2.2.6 Numerical Demonstration

A comparison between Non-Local Means filter with Gaussian filter and Mean filter is presented here, because literature on edge detection almost exclusively uses these two filters as a preprocessing step. To provide a roughly equal basis for comparison, input parameters for Gaussian filter and Non-Local Means filter were adjusted so as to provide a visually similar degree of denoising to Mean filter, whose local window $I(x+t)$ was fixed to 5×5 pixels. This led to a Gaussian filter with standard deviation ρ of 0.75 pixel lengths applied over a discrete 7×7 Gaussian kernel. For Non-Local Mean filter, a source window size of 7×7 with a comparison search restricted to a neighborhood of 31×31 pixels centered at the source pixel was fixed, while noise variance was adjusted until a degree of smoothing visually similar to Gaussian filter and Mean filter was obtained, in practice the noise variance is estimated from existing library.

The comparison of noise-removal techniques for a slice of X-ray tomography image of granular material is done by taking the difference between restored images using Gaussian, Median and NLM filters and the original image as the human eye is not sensitive to discern

the quality of the raw output, Figure 2.1. Top left image shows the raw image slice from reconstructed X-ray image, all other images show the denoised images using different denoising filters and the subtracted noise patterns for respective method in which light color indicates the greatest difference. Grain boundaries can be visually identified from the removed noise by Gaussian filter (Figure 2.1(e)) and Median filter(Figure 2.1(f)), which indicates that these conventional filters remove too much of high frequency details which is crucial to identify the grain-grain and grain-void boundaries. This test case demonstrates that Non-Local Means filter can better preserve the edge sharpness while cleaning up the local intensity variations as shown in Figure 2.1(g).

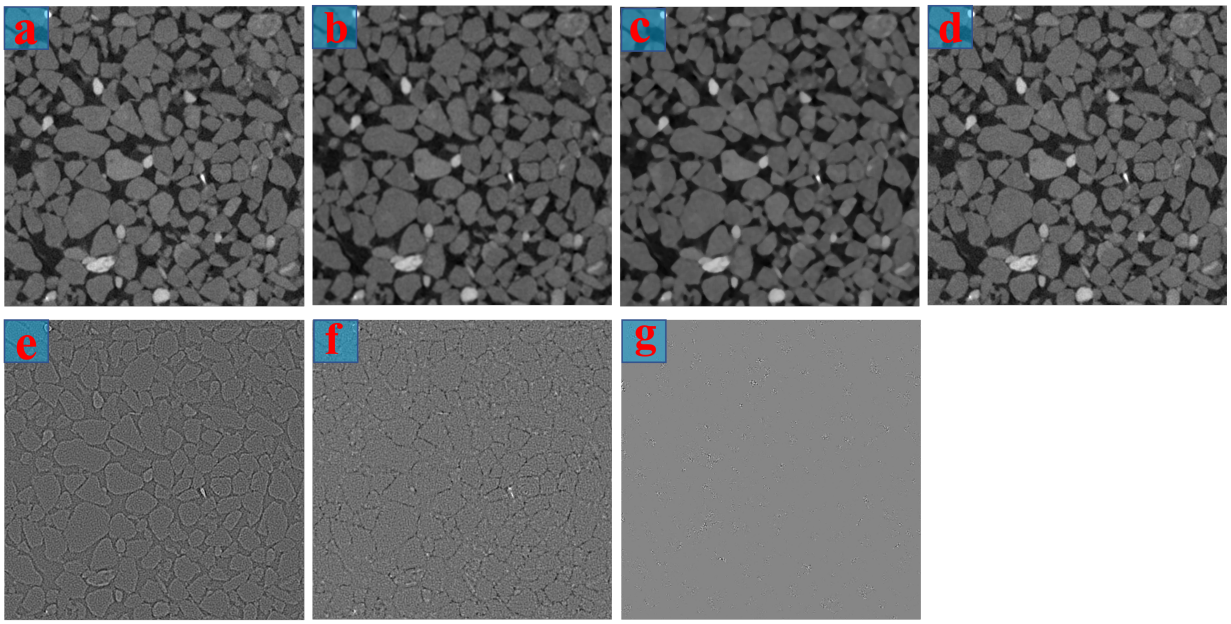

Figure 2.1: (a) Original noised image, (b) Denoised image with Gaussian filter, (c) Denoised image with Median filter, (d) Denoised image with Non-Local Mean filter, (e) Noise removed by Gaussian filter, (f) Noise removed by Median filter, (g) Noise removed by Non-Local Mean filter

Based on the visual indicator shown in Figure 2.1, NLM filter performs extremely well on reducing image noise, particularly for discrete media where it is able to reduce noise within phases while preserving sharpness at both grain-void boundaries and grain-to-grain contacts. Due to the nature of the NLM filter, it is best applied to textured or periodic images which provide enough redundancy to recover original images. This is the case of XRCT images on granular material: for every pixel, we can find enough samples with similar configurations. However, the NLM filter is more computationally expensive method but one could choose the comparison range for windows $I(y+t)$ to speed up the computation as discussed before. Being able to preserve boundaries sharpness is of crucial importance because numerical

studies of mechanical behavior are contingent upon the accuracy of the fabric and grain reconstruction, as both over-smoothing grain-void boundary or deleting small grains would result in a change in the morphological of the grain.

2.3 Image Super-Resolution

The second step in image enhancement is super-resolution. The purpose of this super-resolution step is to increase the native resolution of the images in order to up-scale or improve the details of grain images. Super-resolution techniques were initially developed to overcome the limitations of low-cost hardware such as image sensors, as an image can only be analyzed effectively if its details are visible. Since then, high-resolution images have become available and extremely popular in the field of digital image processing, ranging from satellite imaging to medical imaging. Image super-resolution is viewed as an inverse problem in which the original high-resolution image is recovered using reasonable assumptions or prior knowledge that establishes a link between the high-resolution image and its low-resolution counterpart. This problem is difficult because it is ill-posed, which means that there are not enough low-resolution images, unknown blurring operations, and limited prior knowledge or constraints.

2.3.1 Conventional Approaches

Typical approaches to reconstruction of a high-resolution image are through interpolation from the sub-pixels of several low-resolution images, which can be understood as a generic smoothness prior over the high-resolution image space. Nearest neighbor, bi-linear, bi-cubic interpolation methods are few of the most traditional super-resolution methods. These methods all fail to keep the sharpness at the edges of the output image and leave a visible ringing artifact. Carrato et al. (1996) tried to solve this problem with the idea of edge-sensitive interpolation filter, but it tends to overly emphasize edges in non-homogeneous regions and overly smooth out soft textures. An interpolation method is considered as a "global" procedure because it is also influenced by other intervals, we therefore shall increase the order of polynomial to accommodate an increasing set of data points. From a numerical analysis point of view, classical polynomial interpolation method, e.g., Lagrange interpolation tends to vary rapidly in some part of region of interest and fails to produce smoothly varying function. To solve this problem, cubic spline (Hou & Andrews, 1978) uses piece-wise polynomial to connect knot points to the second order. Compared with linear and cubic interpolation, this method works well for super resolution task as it preserves high-frequency details in images. But it is still known to produce ringing and jagged artifacts. Sun et al. (2008) solved this problem by using gradient field as the image prior for super-resolution, which describes the shape and sharpness of the gradient profile as a distribution and this statistic is mostly stable and uniform regardless of the resolution of the image. However, it tends to produce watercolor-like artifact in the resulting high-resolution images. Chang et al. (2004)

proposed one of the first learning-based methods for image super-resolution, which was inspired by a manifold learning method, Locally Linear Embedding (LLE). LLE-based image super-resolution assumes that image patches in the low-resolution images and corresponding patches in high-resolution images form a similar manifold geometry in two distinct spaces, therefore, high-resolution counterparts can be reconstructed from the low-resolution inputs with the learned parameters. One drawback of this method is it always considers a fixed number of neighbors for learning, making it highly susceptible to blurring due to over/under fitting.

2.3.2 Image Super-Resolution via Sparse Representation

With the popularity of machine learning algorithms, a learning-based method, which uses training data to learn low-to-high-resolution matches, has been highly successful. In this work, the super-resolution method proposed by Yang et al. (2010) was implemented, which utilizes the sparse signal representation to recover high-resolution images from low-resolution counterparts. The idea is inspired from manifold learning that a sparse representation of signal can be correctly captured from its down sampled counterparts However, instead of working directly with the image patch pairs sampled from the high-resolution images, compact over-complete dictionaries are learned for both high and low-resolution images in order to capture the co-occurrence prior and to improve the speed of the algorithm. This approach is motivated by recent results in sparse signal representation, which suggests that the linear relationships among high-resolution signals can be accurately recovered from their low-dimensional projections. Besides, image statistics suggest that image patches can be well-represented as a sparse linear combination of elements from an appropriately chosen over-complete dictionary. Inspired by the above observations, we can seek a sparse representation for each patch of low-resolution input, and then use the coefficients of this representation to generate the high-resolution output. Specifically, two coupled over-complete dictionaries, D_h for high-resolution patches, and D_l for low-resolution patches are concatenated and jointly learned in a probabilistic model. As a result, the sparse representation of a low-resolution patch in terms of D_l can be directly used to recover the corresponding high-resolution patch from D_h.

Compared with an example-based learning strategy that applies to generic images where the low-resolution to high-resolution prediction is learned via a Markov random field through belief propagation (Sun et al., 2003), learning over-complete compact dictionaries requires smaller training patch database and the choice of a feature is no longer critical (Wright et al., 2008). Also, the computation is much more efficient and scalable as it is mainly based on linear programming or convex optimization. Due to the fact that low-resolution image and its high-resolution counterpart should be consistent with each other, the observed low-resolution image Y is considered a blurred and down-sampled version of the high-resolution image X:

$$Y = SHX \tag{2.7}$$

Where H represents a blurring filter and S the down-sampling operator. This is extremely ill-posed problem, and a unique solution is obtained by regularizing the problem via sparsity prior. That is, the high-resolution image patch x can be represented as a sparse linear combination in a dictionary D_h which trained from high-resolution patches sampled from training image:

$$x \approx D_h \alpha, \quad \alpha \in \mathcal{R}^K, \quad \|\alpha\|_0 \ll K \tag{2.8}$$

The sparse representation α is recovered by representing patches y of the input image Y, with respect to a low-resolution dictionary D_l trained together with D_h. A straightforward way to obtain two such dictionaries is to sample image patch pairs directly, which preserves the correspondence between the high-resolution and low-resolution patch items. However, such a strategy would result in large dictionaries and expensive computation. A more effective way is to learn a compact dictionary pair by ensuring both consistency and sparsity requirements via linear programming and convex optimization:

$$\begin{aligned} &D = \arg\min_{D,Z} \|X - DZ\|_2^2 + \lambda \|Z\|_1 \\ &s.t. \quad \|D_i\|_2^2 \leq 1, \quad i = 1, 2, \ldots, K \end{aligned} \tag{2.9}$$

Where the l_1 norm $\|Z\|_1$ is used to enforce sparsity, and the l_2 norm constraints on the columns of D remove the scaling ambiguity. This is one of the most researched topics in optimization and classical machine learning, one common interpretation of constrained linear square problem is formulating it into a MAP problem, and different regularization terms correspond to different priors. i.e., Laplacian prior for l_1 norm and Gaussian prior for l_2 norm. Laplacian prior is better suited to remove spurious oscillations over Gaussian prior which is mathematically simpler and exists in a enclosed form solution (Rudin et al., 1992). Discussion of domain specific usage of image prior was presented by Pickup et al. (2003) and Kim and Kwon (2010). The above equation is not convex in both D and Z at the same time but it is convex in one of them with the other fixed. Solving constrained least square problem separately for D_l and D_h does not produce the same coefficient for high-resolution patch and low-resolution patch. Instead, combining the learning objectives and forcing the high-resolution and low-resolution representations to share the same sparse coefficients. Therefore, the optimization performs in an alternative manner over Z and D:

(1) Initialize D with a Gaussian random matrix, with each column unit normalized.

(2) Fix D, update Z by

$$Z = \arg\min_{Z} \|X - DZ\|_2^2 + \alpha \|Z\|_1 \tag{2.10}$$

This is a LASSO regression formulation and can be solved efficiently through feature-sign search algorithm (Lee et al., 2007).

(3) Fix Z, update D by

$$\begin{aligned} &D = \arg\min_{D} \|X - DZ\|_2^2 \\ &s.t. \quad \|D_i\|_2^2 \leq 1, \quad i = 1, 2, \ldots, K \end{aligned} \tag{2.11}$$

(4) This is a Quadratically Constrained Quadratic Programming (QCQP) that is ready to be solved in many optimization packages.

(5) Iterate between (2) and (3) until converge.

2.3.3 Mathematical Background of Optimization

The second step is a LASSO regularization problem, which helps to maintain the sparsity of solution compared with other problems regularized by a different norm, i.e., ridge regression. The problem of finding the sparsest representation is regularized by the zero-norm by definition, which is in general an NP-hard problem. The use of l_1-norm instead of l_0-norm was justified by Donoho (2006) by showing both the norms produce unique and identical solution, if the desired coefficients are sparse enough. The LASSO regularization problem can be effectively solved with the feature-sign search algorithm, which aims to solve the following equivalent optimization problem:

$$\min_x \|y - Ax\| + \gamma\|x\|_1 \tag{2.12}$$

If we know the signs (positive, zero or negative) of the x_i at the optimal value, we can replace each of $\|x_i\|_1$ term with either x_i (if $x_i > 0$), x_i (if $x_i < 0$), or 0 (if $x_i = 0$). This reduces to a standard, unconstrained quadratic optimization problem, which can be solved analytically and efficiently. The core idea of the feature-sign search algorithm therefore attempts to search or guess iteratively the signs of the coefficients x_i. For each training patch $X[:, i]$, we can apply this algorithm to solve its sparse code $Z[:, i]$ separately, from where Z is obtained as each column of Z is a solution of a l_1 regularized least squares problem. Here a python slice notation is used.

After Z is solved from the second step, the third step involves in solving of another optimization problem over D, which is in the form of:

$$\begin{aligned} &D = \arg\min_D \|X - DZ\|_2^2 \\ &s.t. \quad \|D_i\|_2^2 \leq 1, \quad i = 1, 2, \ldots, n \end{aligned} \tag{2.13}$$

Or equivalently considering the Lagrangian:

$$\mathcal{L}(D, \lambda) = \mathbf{trace}((X - DZ)^T(X - DZ)) + \sum_{i=1}^{n} \lambda_i(\|D_i\|_2^2 - 1) \tag{2.14}$$

Where each $\lambda_j \geq 0$ is a dual variable. Minimizing over D analytically, we obtain the Lagrange dual:

$$D(\lambda) = \min_D \mathcal{L}(D, \lambda) = \mathbf{trace}(X^T - XZ^T(ZZ^T + \Lambda)^{-1}(XZ^T)^T - \Lambda) \tag{2.15}$$

Where $\Lambda = \mathbf{diag}(\lambda)$. Our objective is to solve the Lagrange dual problem with some linear programming solvers like Newton, Conjugate Gradient. To do that, we convert the

constrained least squares problem into unconstrained least squares problem, such that those well-implemented solvers could be used. Three ingredients of those solvers are: objective function, gradient and optionally, Hessian.

(1) Objective function:

$$D(\lambda) = \min_{D} \mathcal{L}(D, \lambda) = \mathbf{trace}(X^T - XZ^T(ZZ^T + \Lambda)^{-1}(XZ^T)^T - \Lambda) \tag{2.16}$$

(2) Gradient:

$$\frac{\partial D(\lambda)}{\partial \lambda_i} = \|XZ^T(ZZ^T + \Lambda)^{-1}e_i\|^2 - 1 \tag{2.17}$$

Where $e_i \in \mathcal{R}^n$ is the i-th unit vector.

(3) Hessian:

$$\frac{\partial^2 D(\lambda)}{\partial \lambda_i \partial \lambda_j} = -2((ZZ^T + \Lambda)^{-1}(XZ^T)^T XZ^T(ZZ^T + \Lambda)^{-1})_{ij}((ZZ^T + \lambda)^{-1})_{ij} \tag{2.18}$$

(4) Since then, the Lagrange dual problem can be optimized using conventional solvers, i.e., Newton's method or Conjugate Gradient. The optimum dictionary D is obtained via:

$$D^T = (ZZ^T + \lambda)^{-1}(XZ^T)^T \tag{2.19}$$

2.3.4 Numerical Demonstration

XRCT image of granular material is an excellent research candidate because it has highly repetitive, regular patterns, hence it only requires a small batch of training images. The super-resolution results obtained by sparse signal representation of XRCT images of granular material are demonstrated using a patch size as 5×5 pixels for both low-resolution and high-resolution images. The two dictionaries for high-resolution and low-resolution image patches are trained from 10000 patch pairs randomly sampled from an XRCT dataset; the low-resolution images were generated via bi-cubic interpolation using high-resolution counterparts. For high-resolution images, the image patches were flattened into a $(25, 1)$ vector; for low-resolution images, feature extractor filters, i.e., gradient and Laplacian were first applied in both horizontal and vertical direction to extract information at high-frequency textured regions. These four features were concatenated with original patch and produced a feature vector of size $(100, 1)$. When determining the dictionary size, we found that 512 is sufficient for our application but intuitively a larger dictionary possess more expressive power yielding more accurate approximation even though it costs more in computational power. Therefore, the input training data is $X \in R^{125\times10000}$. The dictionary size is initialized as $D \in R^{10000\times512}$ to sparsely encode image patches with 512 atoms while this number might be smaller in the end because columns with zero norm would be abandoned. The

choice of regularization parameter α depends upon the level of noise in the input image, the noisier the data, the larger the value of α should be, for this experiment, setting $\alpha = 0.2$ generally yields satisfactory results. The results were evaluated both via visual inspection and qualitative Root Mean Square Error (RMSE), which compares the differences between ground truth high-resolution images and the recovered images. With N is the number of pixels in images I_1 and I_2.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(I_{1,i} - I_{2,i})^2}{N}} \tag{2.20}$$

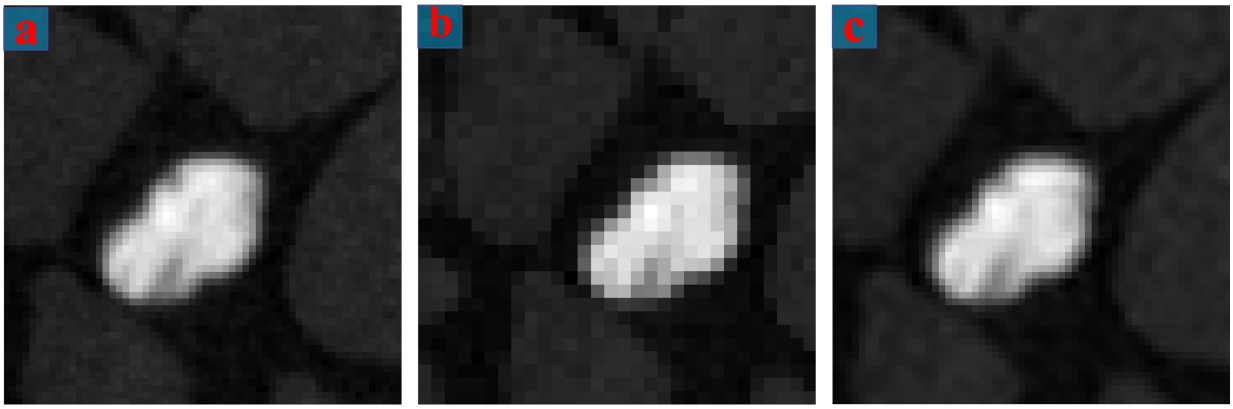

Figure 2.2: Performance of super-resolution with sparse representation (a) high-resolution ground truth; (b) low-resolution counterpart derived from (a); (c) recovered high-resolution image with sparse representation.

As shown in Figure 2.2, the sparse representation method has exceptional performance over the more traditional methods as there are visually no differences between the ground truth and the recovered image. The descriptor RMSE is used to quantify the enhancement, compared with conventional methods, namely nearest neighbor, bi-linear, bi-cubic, which has RMSE score 5.80, 4.81 and 5.10. The super-resolution technique based on sparse representation only has 2.98 RMSE.

2.4 Image Binarization

The next major step in this process is to segment the enhanced images by labeling each pixel whether it belongs to the grain phase or to the void phase. This is not a difficult job for clean and coarse granular material, as it is possible to highlight different components in the object being imaged and produce high-contrast piece-wise constant images, which greatly

facilitate image segmentation. However, ideal imaging conditions are not always obtainable in practice. The XRCT quality is degraded considerably by electronic noise, lighting conditions, and partial-volume effect, all of which cause the intensity histogram of the different components to overlap. And this is especially common for XRCT on naturally deposited granular material where the object is often "polluted" by with finer impurities. In this case, the spatial intensity inhomogeneities are often of sufficient magnitude to cause distributions of signal intensities associated with grain and void classes to overlap significantly. In addition, operating conditions, experiment setup and status of X-ray equipment frequently affect the observed intensities, causing significant inter-scan intensity inhomogeneities which necessitate manual adjustment on a per-scan basis.

2.4.1 Conventional Methods

Even the most naive and straightforward binarization technique works reasonably well for high-contrast piece-wise constant images, where a global threshold value is manually selected such that pixels with high intensity levels are classified as grains and void otherwise. If this method were to be applied for a more challenging task, i.e., naturally deposited granular material, it might not produce a unique and satisfactory outcome as this manner of selecting threshold is heavily influenced by selector bias. It may not be optimum, without any prior knowledge, to select a hard decision boundary from the intensity histogram where distributions of different clusters overlap. Hence the output from the binarization process is highly likely to be different for different runs, making the process questionable.

To overcome this issue and to make the process more robust and automatic, statistical based methods like k-mean clustering and Otsu's method are widely used (MacQueen et al., 1967; Otsu, 1979). In those methods, the threshold value was calculated by maximizing the variance among clusters while minimizing the variance within each cluster. Take Otsu's method as an example, it is widely used in preliminary image processing because it is robust and simple. This method segments gray images with two object classes and later it was extended to process more than two classes, which bears the same spirit as the k-mean cluster algorithm. The fundamental idea of Otsu's method is to search for the best threshold value t such that:

$$\arg\min_{t} = \sigma_{\omega}^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t) \tag{2.21}$$

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i) \tag{2.22}$$

$$\omega_1(t) = \sum_{i=t}^{L-1} p(i) \tag{2.23}$$

$$p(i) = \frac{n_i}{N} \tag{2.24}$$

Where $\sigma_\omega^2(t)$ is the inter-class variance of the two clusters, $\sigma_0^2(t)$ and $\sigma_1^2(t)$ are the intra-class variance of the two clusters with t being the global threshold value to hard binarizing image into two clusters. $\omega_0(t)$ and $\omega_1(t)$ represents the probability of class occurrence and $p(i)$ is the probability of i-th intensity level with N being the total number of pixels in the image.

In all, this type of method labels the pixels according to their probability values, which are determined based on the intensity distribution of the image. Therefore, with a suitable assumption about the distribution and given only the intensity for each pixel, statistical approaches can go one step beyond hard threshold approaches and formulate an estimation problem with some criterion. Maximum a posterior (MAP) and maximum likelihood (ML) principles are two of such examples. For example, the formulation of density function of the pixel intensity is often chosen to be mixture Gaussian model (MGM) (Wells et al., 1996; Guillemaud & Brady, 1997) because this model connects MAP and ML in an elegant way and the formulation is mathematically simple. In segmenting XRCT images of granular material which contains two phases, the actual shape of histogram is also very similar to the Gaussian as shown in Figure 2.3. This is the result of large number theorem and central limit theorem: the number of pixels in an image is usually large enough such that the probability density function of pixel intensity can be captured as Gaussian.

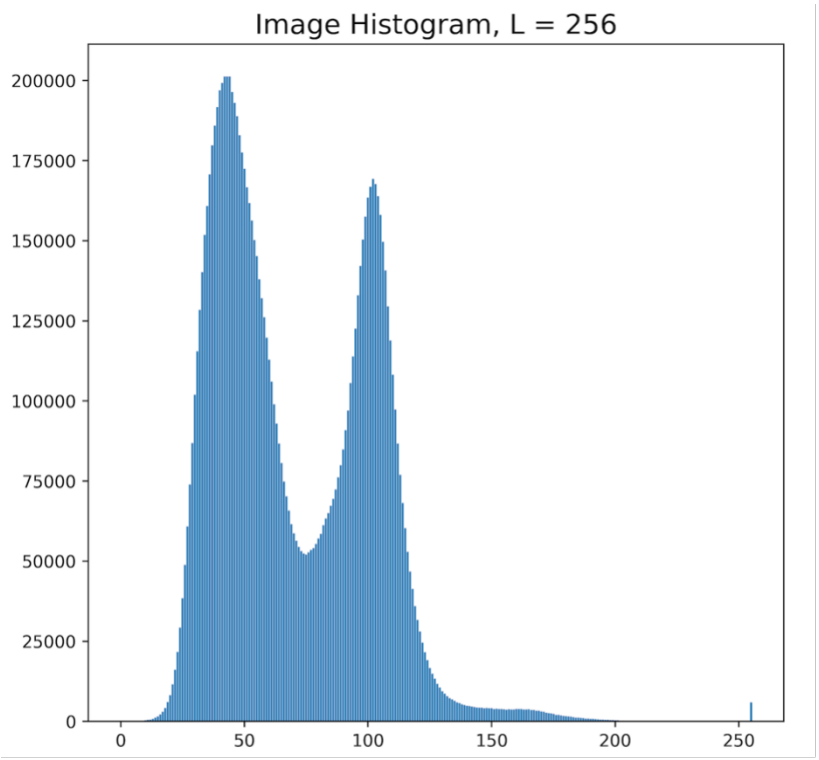

Figure 2.3: Intensity histogram of typical XRCT images resembles Mixture Gaussian.

2.4.2 Hidden Markov Random Field (HMRF)

However, because all of the models presented above are intensity-based, they all have an inherent limitation: spatial information is ignored because each pixel is independently sampled from a distribution. Due to this limitation, purely statistical models can only be used on well-defined images with low noise levels, and their performance is degraded by artifacts such as the partial volume effect and bias field distortion. A model that depicts the relationship between sequences via probabilities, the Markov random field (MRF) is particularly well-suited to addressing this issue. Given that our objective is to infer the pixel class via a field of observation, and that the true labels cannot be observed directly, the hidden Markov random field (HMRF) is used to exploit spatial information in such a way that the image is encoded using the contextual constraints of neighboring pixels, as illustrated in Figure 2.4. Such constraints could be as simple as requiring that each pixel have the same class labels as its neighbors, as the clustered output is piecewise constant by nature. This is accomplished by prioritizing mutual influences between pixels in the supplement mixture Gaussian model.

For a given image $y = (y_1, y_2, \ldots, y_N)$ with y_i representing the observed intensity level of $i - th$ pixel, our goal is to deduce a connection between underlying label configuration and observed image intensity field. That is to infer the corresponding set of labels $x = (x_1, x_2, \ldots, x_N)$ where $x_i \in L$,and for binary segmentation problem $L = \{0, 1\}$. HMRF is inherently a probabilistic model. Since each cluster in the actual histogram distribution resembles Gaussian distribution, we still use MGM to map the generative relationship between intensity level of a pixel given its cluster label. In fact, using MGM alone essentially embraces the idea of MLE principle and this is the basis of the aforementioned statistical approaches, but the decision boundary between grain and void is softer compared to global threshold approaches like Otsu's method. If we were to accommodate the spatial information as prior, we can use the MAP criterion, assuming that pixels are more likely to belong to the same cluster as their neighbors. Hence, the optimum configuration of labels can be formulated and solved as a discrete optimization problem:

$$x^* = \arg\max_x \{P(y|x, \Theta)P(x)\} = \arg\max_x \{P(x, y|\Theta)\} \tag{2.25}$$

Where $\Theta = \{\theta_l | l \in L\}$ is the parameter space, $P(x)$ is the prior distribution of labels which implicitly encoded through HMRF, $P(y|x, \Theta)$ is the conditional probability of an intensity level if we knew its class.

2.4.3 Expectation-Maximization (EM) Algorithm

Additionally, we must determine the parameter space to complete the HMRF model. As a result, the model parameters are determined using the EM algorithm. The EM algorithm is widely used in statistics as an iterative method for parameter identification in statistical models, particularly when the model is dependent on unobserved latent variables. It has

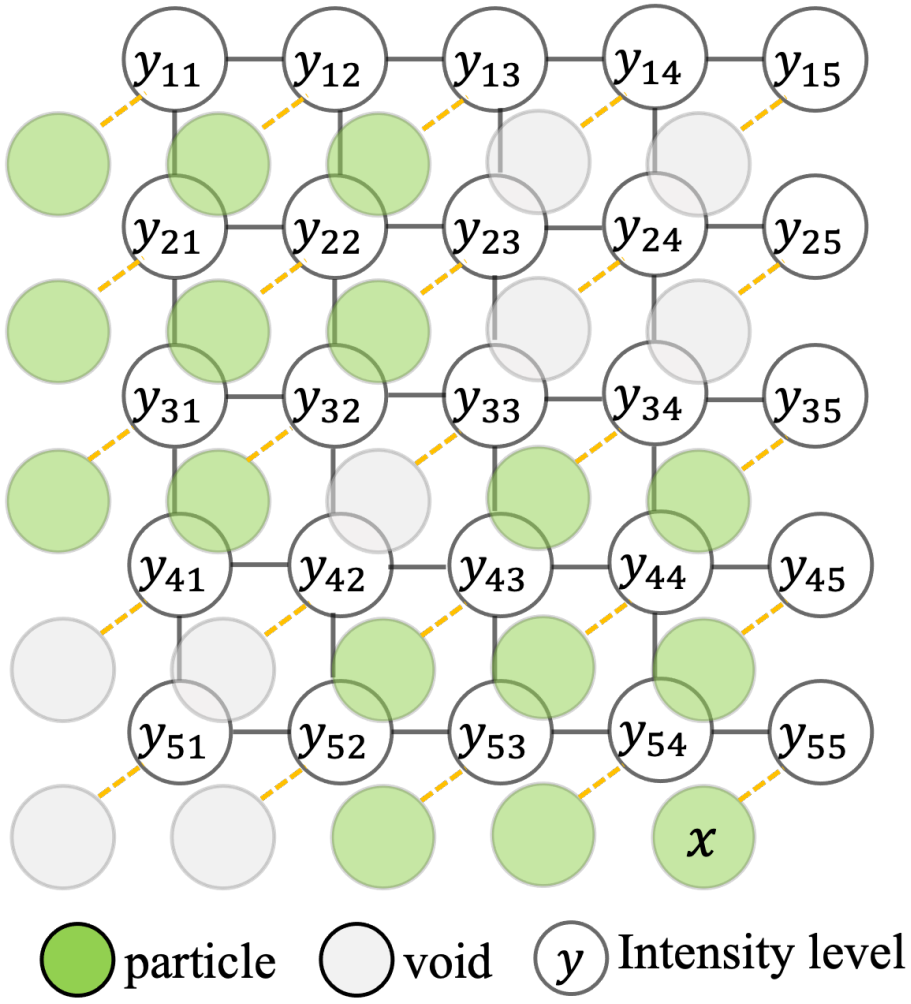

Figure 2.4: Demonstration of HMRF, pixel labels are inferred through a field of observation, which is the intensity level of XRCT images.

been demonstrated to be more suitable for this type of problem than gradient descent-based approaches, as arbitrary and discrete partitions provide no gradient information in a clustering problem. This algorithm iterates alternately between two steps. The Expectation step (E-step) generates an optimization formula for the free energy, which is a function of expected complete log likelihood, using parameters pre-calculated in the previous step or initialized, and the Maximization step (M-step) computes parameters that maximize the free energy in the E-step. The following summarizes the mechanism of EM:

(1) Start: Initialize parameter set Θ^0.

(2) The E-step: Compute posterior probability $P(x_i = l|y_i, \Theta^t)$ for each observed pixel y_i with respect to parameters of l-th cluster, which is the first and second moment of inertial

μ_l and Σ_l in the MGM model. Mathematically, E-step seeks a distribution q such that:

$$q^{t+1} = \arg\max_q F(q^t, \Theta^t) = P(x|y, \Theta^t) \tag{2.26}$$

Where $F(q^t, \Theta^t)$ is the free energy at t-th iteration, it is a function of the expected complete log likelihood and the entropy of distribution q. $F(q^t, \Theta^t)$ is the lower bound of log likelihood $\log P(y_i)$ for each pixel, and the bound is obtainable when q is the posterior of label x_i with respect to the intensity level y_i. Sketch of proof:

$$\begin{aligned}
\log P(y_i) &= \log \sum_{l=1}^{L} P(y_i, x_i = l|\Theta) = \log \sum_{l=1}^{L} \frac{q(x_i|y_i,\Theta)P(y_i, x_i|\Theta)}{q(x_i|y_i,\Theta)} \\
&= \log \sum_q \frac{P(y_i, x_i|\Theta)}{q(x_i|y_i,\Theta)} \geq \sum_q \log \frac{P(y_i, x_i|\Theta)}{q(x_i|y_i,\Theta)} \\
&= \sum_{l=1}^{L} q(x_i|y_i,\Theta) \log P(y_i, x_i|\Theta) - \sum_{l=1}^{L} q(x_i|y_i,\Theta) \log q(x_i|y_i,\Theta) = F(q,\Theta)
\end{aligned} \tag{2.27}$$

(3) The M-step: Maximize free energy $F(q^{(t+1)}, \Theta^t)$:

$$\Theta^{t+1} = \arg\max_\Theta F(q^{t+1}, \Theta^t) = \arg\max_\Theta \sum_{l=1}^{L} q(x_i|y_i,\Theta) \log P(y_i, x_i|\Theta) \tag{2.28}$$

(4) Repeat the E-step and the M-step until convergence.

One significant conclusion is that maximization of free energy in the E-step requires posterior probability over latent variable.

2.4.4 Weighted Mixture Gaussian Model

We can also consider that pixels in the different clusters might be weighted differently, e.g., certain clusters are more important than others. Let $w > 0$ a weight indicating the relative importance of the observation intensity of a pixel y_i. Intuitively, a larger value of w implies a stronger influence towards y_i. As for the Gaussian distribution, the multiplicity w can be interpreted as observing y_i w times, and the likelihood function becomes:

$$\hat{P}(x|\Theta, w) = \mathcal{N}(x; \mu, \frac{1}{w}\Sigma) \tag{2.29}$$

From which the MGM with L components become:

$$\tilde{P}(x; \Theta, w) = \sum_{l=1}^{L} \pi_l \mathcal{N}(x; \mu, \frac{1}{w_l}\Sigma_l) \tag{2.30}$$

Where $\Theta = \{\pi_1, \ldots, \pi_L, \mu_1, \ldots, \mu_L, \Sigma_1, \ldots, \Sigma_L\}$ are the mixture parameters, $\pi_1, \ldots, \pi_L$ are the prior probability satisfying $\pi_l \geq 0$ and $\sum_{l=1}^{L} \pi_l = 1$; $\{\mu_l, \Sigma_l\}$ are the parameters of the l-th class. It is straightforward to show that the weighted MGM yields following EM algorithm:

(1) The E-step computes the posterior probability of pixel y_i from cluster l with parameter $\{\mu_l, \Sigma_l\}$ and weight w_i.

$$q_{il}^{t+1} = \frac{\pi_l^t \hat{P}(y_i; \mu_l, \Sigma_l, w_i)}{\tilde{P}(y_i; \Theta, w_i)} \tag{2.31}$$

where $\hat{P}$ and $\tilde{P}$ are defined before

(2) the M-step updates parameters space by maximizing free energy:

$$\begin{aligned} \Theta^{t+1} &= \arg\max_{\Theta} \sum_{i=1}^{N} \sum_{l=1}^{L} q_{il}^{t+1} \log \pi_l \mathcal{N}(y_i; \mu_l, \frac{\Sigma_l}{w_i}) \\ &= \arg\max_{\Theta} \sum_{i=1}^{N} \sum_{l=1}^{L} q_{il}^{t+1} (\log \pi_l - \log |\Sigma_l|^{\frac{1}{2}} - \frac{w_i}{2} (y_i - \mu_i)^T \Sigma_l^{-1} (y_i - \mu_i)) \end{aligned} \tag{2.32}$$

By canceling out the derivatives with respect to the model parameters, we obtained the following update formula for the mixture proportions, means and covariance matrices:

$$\pi_l^{t+1} = \frac{1}{N} \sum_{i=1}^{N} q_{il}^{t+1} \tag{2.33}$$

$$\mu_l^{t+1} = \frac{\sum_{i=1}^{N} w_i q_{il}^{t+1} y_i}{\sum_{i=1}^{N} w_i q_{il}^{t+1}} \tag{2.34}$$

$$\Sigma_l^{t+1} = \frac{\sum_{i=1}^{N} w_i q_{il}^{t+1} (y_i - u_i^{t+1})(y_i - u_i^{t+1})^T}{\sum_{i=1}^{N} q_{il}^{t+1}} \tag{2.35}$$

2.4.5 Posterior Free Energy

The image classification problem involves assigning to each pixel, characterized by an intensity value y_i, a class label taking a value from the set L. The problem of classification is the problem of recovering x^* from an observed image y. Here x^* is the true but unknown labeling configuration and $\hat{x}$ is an estimate of x^*, both of which are interpreted as particular realizations of a configuration field. According to the MAP criterion:

$$\hat{x} = \arg\max_{x \in \mathcal{X}} \{P(y|x)P(x)\} \tag{2.36}$$

In MGM, the pixel intensity y_i follows a Gaussian distribution with parameters $\{\mu_l, \Sigma_l\}$, given the class label $x_i = l$.

$$P(y_i|x_i) = \frac{1}{\sqrt{2\pi^d|\Sigma_l|}} \exp\left(-\frac{(y_i-\mu_i)^T\Sigma_l^{-1}(y_i-\mu_i)}{2}\right) \tag{2.37}$$

$$\begin{aligned} P(y|x) &= \prod_{i=1}^{N} P(y_i|x_i) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi^d}} \exp\left(-\frac{(y_i-\mu_i)^T\Sigma_l^{-1}(y_i-\mu_i)}{2} - \frac{\log|\Sigma_l|}{2}\right) \\ &= \frac{1}{Z'} \exp\left(-U(y|x)\right) \end{aligned} \tag{2.38}$$

With the likelihood energy:

$$U(y|x) = \sum_{i=1}^{N} U(y_i|x_i) = \sum_{i=1}^{N} \frac{(y_i-\mu_i)^T\Sigma_l^{-1}(y_i-\mu_i)}{2} + \frac{\log|\Sigma_l|}{2} \tag{2.39}$$

And $Z' = (2\pi)^{Nd/2}$ is the normalization constant and d is the dimension of random variable. In our problem, d=1 as we only study the intensity levels of image pixels which is a scalar. When the joint probability density of the random variables is strictly positive, every Markov random field can be presented by a Gibbs distribution as the consequence of Hammersley-Clifford theorem. Therefore, the context-dependent spatial information can be modeled as a prior distribution in the form $P(x) = \frac{1}{Z}\exp\left(-U(x)\right)$, where $Z = \sum \exp\left(-U(x)\right)$ and$U(x)$ is the prior energy function which has the form:

$$U(x) = \sum_{c\in\mathcal{C}} V_c(x) \tag{2.40}$$

Where $V_c(x)$ is the clique potential and $\mathcal{C}$ is the set of all possible cliques. In the image domain, we assume that one pixel has at most 4 neighbors for a 2D image and at most 6 neighbors for a 3D image. Then the clique energy is defined on pairs of neighboring pixels:

$$V_c(x_i, x_j) = \frac{1}{2}(1 - I_{x_i,x_j}) \tag{2.41}$$

Where $I_{x_i,x_j} = \delta_{ij}$, when $x_i = x_j$, $I_{x_i,x_j} = 1$ and 0 otherwise. Specifically, x^0 is obtained with fundamental image binary segmentation methods such as Otsu's method, K-means or histogram analysis, which becomes the initial guess of HMRF algorithm. It is easy to show that recovering the underlying labelling configuration x^* is equivalent to minimize total free energy.

2.4.6 Numerical Demonstration

Various methods, including Otsu's method, K-means method, and HMRF algorithm, have been explored to binarize typical XRCT images on granular material. The first experiment shows the binarization results on high-resolution, low histogram contract dataset as shown in Figure 2.5. The resulting binarization images and corresponding watershed labels using the aforementioned three methods are shown in Figure 2.6. There are visually no differences between Otsu's method and the K-means method, and both methods produce a similar number of labeled grains after watershed algorithm. Both methods suffer the isolated island effect: a single pixel inside the grain phase was mis-classified as the void pixel, hence substantially changes the Euclidean distance mapping in watershed process. As a result, one larger grain can be incorrectly divided into several smaller ones. This is shown in blue square in (d) and (e) of Figure 2.6. Another issue with these two methods is that: they did not utilize spatial information and could be overly sensitive to local intensity variance and produced fragmented grains, one instance is shown in red square. Both issues were alleviated when the HMRF algorithm is used because the spatial configuration was taken as a prior, which made the binarization processes less sensitive to local intensity noises. Consequently, the HMRF produced less fragmented grains. The convergence rate was studied in Figure 2.7, the initial clustering configuration was generated from Otsu's method and K-means method, respectively, and they displayed similar posterior free energy at the beginning and converged to the same optimum value after 10 iterations.

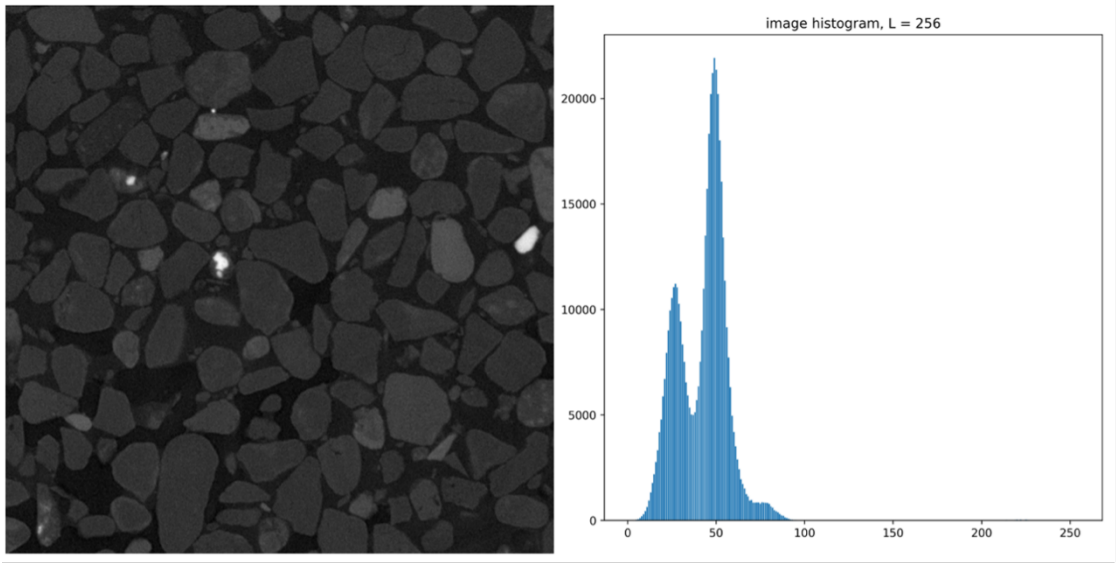

Figure 2.5: Left: High-resolution XRCT images on granular material. Right: Image histogram shows that two clusters have close peaks.

The second experiment shows the binarization results on low-resolution XRCT images as shown in Figure 2.8, Otsu's method and HMRF algorithm are used for comparison. In this

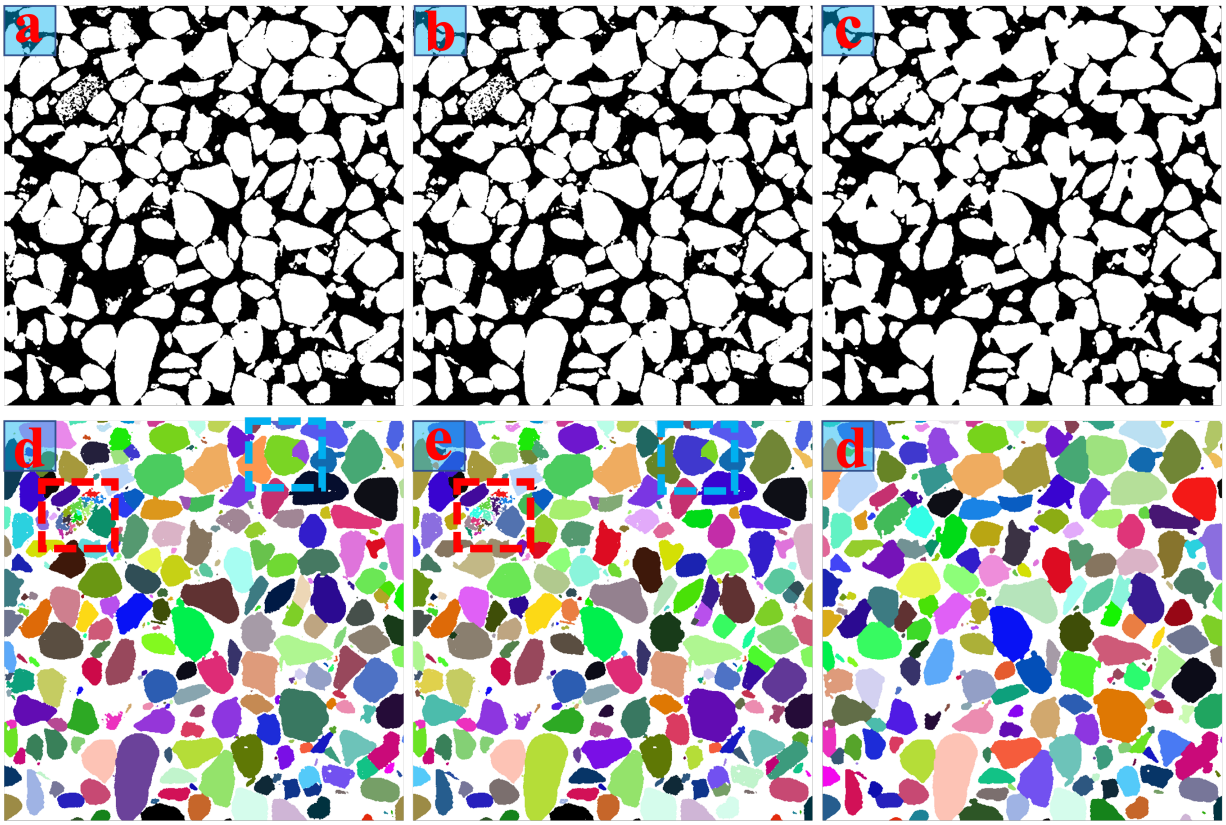

Figure 2.6: Binarized image using (a) Otsu's method, (b) K-means method, (c) HMRF-EM method. Watershed labelled image from (a) Otsu's method, 289 grains, (b) K-means method, 284 grains, (c) HMRF-EM method, 229 grains.

case, there are visually little differences between those two methods and the number of labeled grains generated are similar. The HMRF algorithm seems to have a very small amount of morphological dilation effects on the binarized images, this is because the spatial constraints tend to classify pixels into the same cluster as its neighbors. This effect became pronounced in low-resolution scans because of the partial volume effects between grain-void boundaries, but less salient for better resolution data. Overall, the HMRF algorithm produced fewer grains because it aggregates several grains into unrealistically large grain, while Otsu's method and K-means method failed to capture the true configuration either. As shown in the red square of Figure 2.8, Otsu's method produced an isolated pixel inside a grain and the watershed algorithm divided a large grain into several ones afterward. If the labeled images (d) and (e) in Figure 2.8 were compared with the ground truth (b), both are wrong. In the context of image binarization based on the grey level intensity and spatial constraints, the local region marked in the red square should be considered as a whole because the corresponding

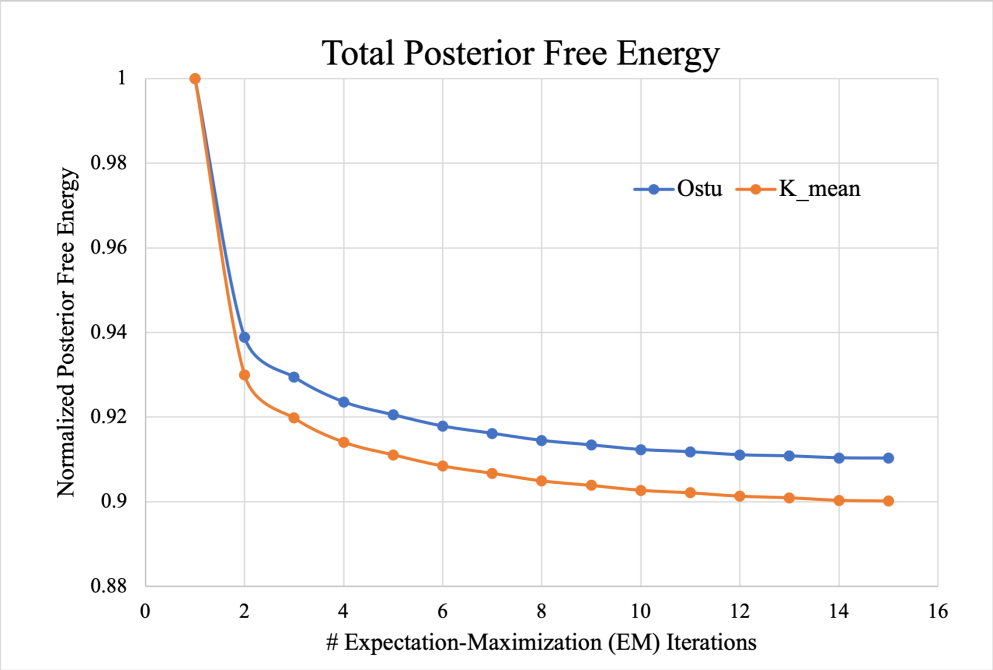

Figure 2.7: Total posterior energy decreases with increasing iteration.

probability map will vote for the grain phase, which overweighs the influence of intensity variance. Sometimes, the HMRF algorithm can make some remedies to fix this problem by assigning void pixels a higher relevance w, and a comparison is made between binarized images using different value of w is shown in Figure 2.9, but the improvement is limited. Intuitively, smaller w implies void pixel is considered less important than grain pixel, this can be interpreted as each void pixel is evaluated w times due to the formulation of weighted EM algorithm. In the demonstration, w took values 0.5, 1.0, 2.5, and 5.0, as w increases, more and more pixels are classified as void, this parallelizes the morphological erosion operation but based on a statistic model. When w became too large, the HMRF algorithm could not correctly label pixels as shown in the red squares of Figure 2.9(d). This is because we assumed intensity distribution obeys MGM in the HMRF algorithm and labelled pixel according to the MAP criterion. When w is large, the covariance $\frac{\Sigma}{w}$ of the void phase tends to spread across a wider region and displays a fat tail, this results in the very bright pixel having higher probability to be void rather than grain. As shown in the Figure 2.5, the XRCT image contains few bright spots, and the intensity histogram has a third peak due to those cluster of pixels with high intensity. Hence the fundamental assumption of MGM, HMRF method may not be able to fully capture the probability distribution of intensity histogram, especially when the low-resolution, noisy images are taken as input.

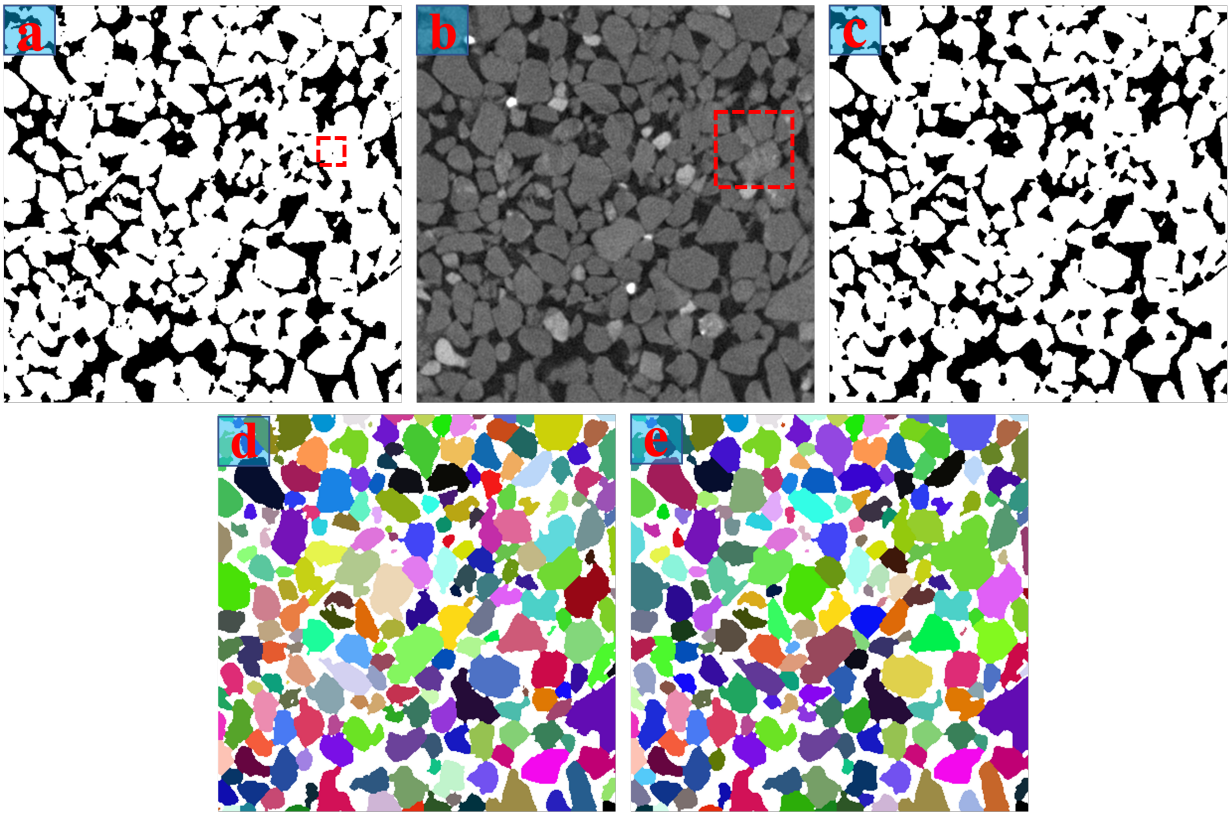

Figure 2.8: Binarized image using (a) Otsu's method, (b) Low-resolution XRCT image, (c) HMRF-EM method. Watershed labelled image from (d) Otsu's method, 217 grains, (e) HMRF-EM method, 210 grains, void importance $w = 2.5$.

2.5 Morphology Reconstruction Using the LS Algorithm

The final step in this workflow is to extract information about grain morphology and to characterize grain surfaces using the LS function. Clearly, the output is highly correlated with the XRCT image quality. As proposed by Osher and Sethian (1988), the concept of using a LS to represent the shape of an object has garnered considerable attention in the field of image segmentation because it is capable of accurately capturing the complex morphology of natural granular material. The advantage of LS algorithms is that they can track the motion of complex topology changes on a fixed Eulerian grid, which is useful for dealing with topological changes as the curve evolves. The fundamental idea behind LS-based image segmentation is to implicitly represent an object's boundaries in a higher-dimensional space via grid-based interpolation. Although not explicitly stated, the signed distance function

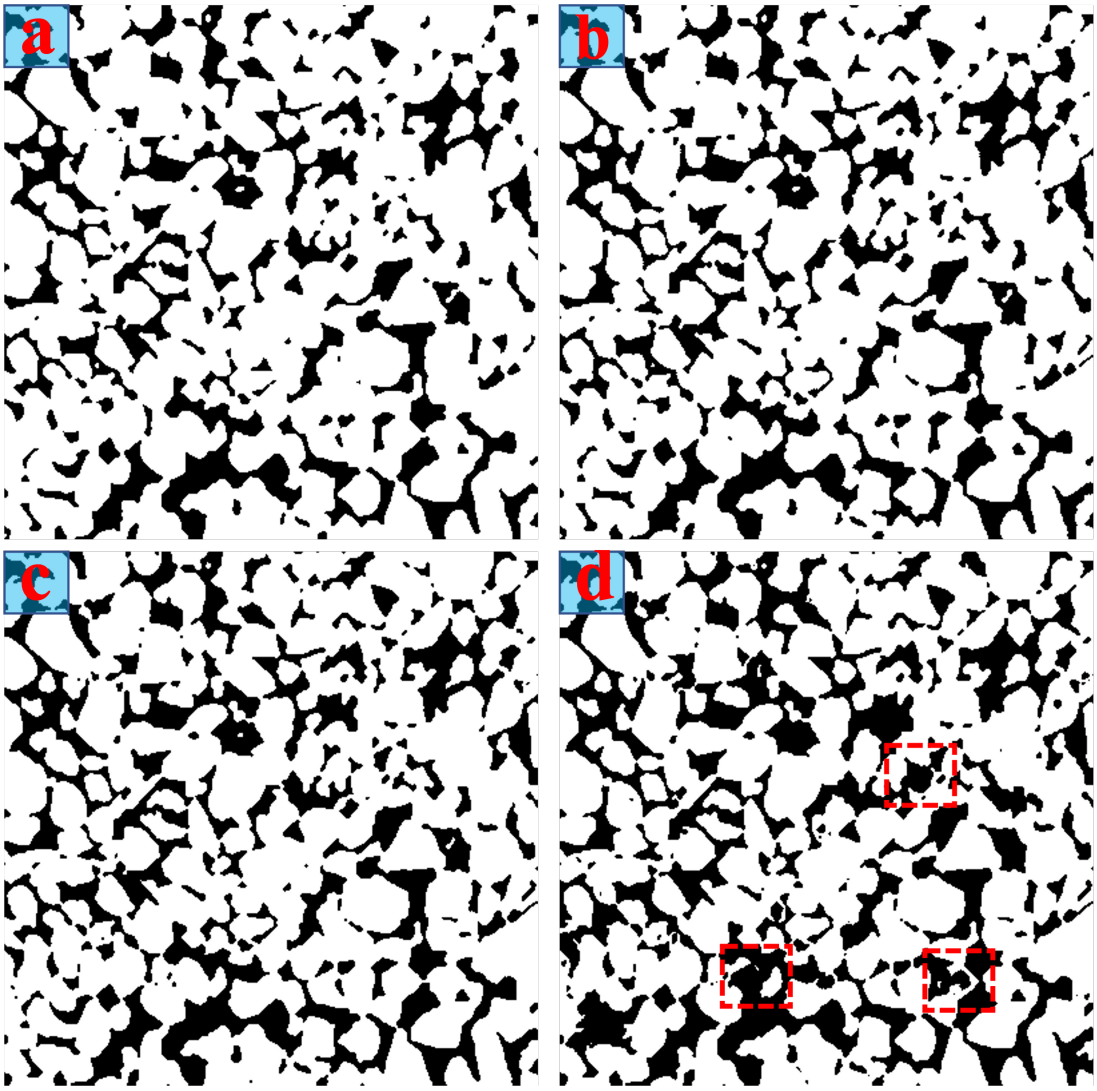

Figure 2.9: Binarized images using HMRF with different void relevance (a) $w = 0.5$; (b) $w = 1.0$; (c) $w = 2.5$; (d) $w = 5.0$.

is used by the majority of LS-based algorithms to generate nonzero (positive or negative) LS values with physical significance: the signed distance between the point of interest and the nearest boundary. Additionally, it stabilizes and facilitates LS evolution convergence. Figure 2.10 illustrates how to represent the grain shape using a LS algorithm. However, conventional LS models do not require compatibility between the signed distance function and the LS function, and the LS function must be re-initialized periodically throughout the evolution to ensure stable results, making the entire process computationally expensive. Li et al. (2010) address this issue in their distance regularized LS algorithm. This algorithm

incorporates an internal energy term into the LS variational formulation, penalizing the LS function for deviating from a signed distance function; this is referred to as DRLSE. One straightforward internal energy term for enforcing the signed distance property is as follows:

$$\mathcal{R}(\phi) = \frac{1}{2}\int_{\Omega}(\|\nabla\phi - 1\|)^2 d\Omega \tag{2.42}$$

Which characterizes the deviation of ϕ from the signed distance function ($\|\nabla\phi\| = 1$). The energy functional $\mathcal{R}(\phi)$ is therefore regularized the entire domain to maintain signed distance property. The LS method aims to implicitly encode the closed surface of object as the zero LS, and the value of LS function ϕ is assumed to take negative values inside the region delimited by the enclosed surface, and positive values outside. Different sign convention is found in other literature such as Lai and Chen (2019).

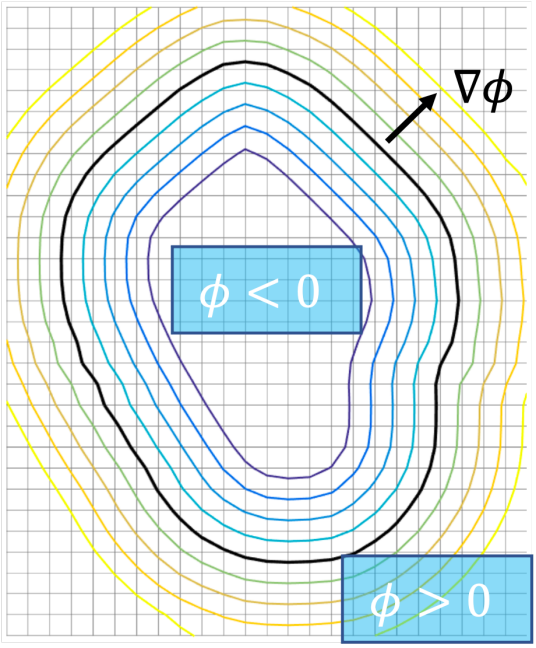

Figure 2.10: Grain shape is implicitly represented by LS function (modified from Kawamoto (2018)).

2.5.1 LS Curve Evolution

The objective of image segmentation is about finding the closed surface for the object in interest, which is nothing more than locating the zero LS:

$$\Gamma = \{(x, y) \in \Omega | \phi(x, y) = 0\} \tag{2.43}$$

Where ϕ is the LS function; (x, y) is the spatial coordinate (pixel position on the image); Ω is the domain of interest and Γ is the closed surface (grain-void boundary). When applying the LS method to identify the particle boundary, the LS function ϕ is evolved by minimizing an appropriately formulated energy functional with an internal distance regularization term and an external energy (stimulus of input images) term that drives the motion of the zero LS towards to the desired locations. The general DRLSE (Distance Regularized Level Set Evolution) formulation can be used in various applications with different definitions of external energy. For image segmentation applications, a variety of image information, including region-based or edge-based image formation, can be used to define the external energy. In terms of the internal energy term, an edge indicator function g is defined, which usually takes smaller values at object boundaries.

$$g = \frac{1}{(1 + \|\nabla G_\rho * I\|^2)} \tag{2.44}$$

Where G_ρ is a Gaussian kernel with a standard deviation σ. In this work, the following functional formulation used by Lai and Chen (2019) is adopted, which is slightly simplified from the original DRLSE in the distance regularization term $\mathcal{R}(\phi)$ (Li et al., 2010):

$$\mathcal{F}(\phi) = \mu\mathcal{R}(\phi) + \lambda\mathcal{L}(\phi) + \nu\mathcal{A}(\phi) \tag{2.45}$$

$$\mathcal{R}(\phi) = \frac{1}{2}\int_\Omega (\|\nabla\phi\| - 1)^2 d\Omega \tag{2.46}$$

$$\mathcal{L}(\phi) = \int_\Omega g(I)\delta(\phi)\|\nabla\phi\| d\Omega \tag{2.47}$$

$$\mathcal{A}(\phi) = \int_\Omega g(I)H(-\phi) d\Omega \tag{2.48}$$

Where $\lambda > 0$ and $\mu, \nu \in R$ are the coefficients of the energy functionals $\mathcal{L}(\phi)$ and $\mathcal{A}(\phi)$, $\delta(\phi)$ and $H(\phi)$ are Dirac delta function and the Heaviside function and are defined below, respectively.

$$\delta_\epsilon(x) = \begin{cases} \frac{1}{2\epsilon}(1 + \cos\frac{\pi x}{\epsilon}) & |x| \le \epsilon \\ 0 & |x| > \epsilon \end{cases} \tag{2.49}$$

$$H_\epsilon(x) = \begin{cases} \frac{1}{2}(1 + \frac{x}{\epsilon} + \frac{1}{\pi}\sin\frac{\pi x}{\epsilon} & |x| \le \epsilon \\ 1 & x > \epsilon \\ 0 & x < -\epsilon \end{cases} \tag{2.50}$$

The LS regularization term $\mathcal{R}(\phi)$ is referred to as a distance regularization term for its role of maintaining the signed distance property of LS function in the entire domain. With the Dirac delta function δ, the energy $\mathcal{L}(\phi)$ computes the line integral of the edge indicator g

along the zero-level contour of ϕ. The energy $\mathcal{L}(\phi)$ is minimized when the zero-level contour of ϕ is located at the object boundaries, this idea is first introduced by Caselles et al. (1997) in their proposed geodesic active contour. The energy functional $\mathcal{A}(\phi)$ computes a weighted area of the region has negative LS value (grain phase). If the initial contour is placed outside the object, the coefficient ν in weighted area term should be positive, so that the zero-level contour can shrink in the LS evolution. If the initial contour is placed inside the object, the coefficient ν should take negative value to expand the contour.

2.5.2 Removal of Anomalies from Reconstruction

A standard method to minimize an energy functional $\mathcal{F}(\phi)$ is to find the steady state solution of the gradient flow equation, which leads to the associated Euler-Lagrange equation or Gateaux derivative of ϕ (Aubert et al., 2006). This is an evolution of a time-dependent function $\phi(x,t)$ in the steepest descent direction of the Gateaux derivative:

$$\frac{\partial \phi}{\partial t} = -\frac{\partial \mathcal{F}}{\partial \phi} = -(\mu \frac{\partial \mathcal{R}}{\partial \phi} + \lambda \frac{\partial \mathcal{L}}{\partial \phi} + \nu \frac{\partial \mathcal{A}}{\partial \phi}) \tag{2.51}$$

$$\frac{\partial \mathcal{R}}{\partial \phi} = \nabla^2 \phi - \nabla \cdot \frac{\nabla \phi}{\|\nabla \phi\|} \tag{2.52}$$

$$\frac{\partial \mathcal{L}}{\partial \phi} = \delta\phi(\nabla g(I) \cdot \frac{\nabla \phi}{\|\nabla \phi\|} + g(I)\nabla \cdot \frac{\nabla \phi}{\|\nabla \phi\|}) \tag{2.53}$$

$$\frac{\partial \mathcal{A}}{\partial \phi} = g(I)\delta(\phi) \tag{2.54}$$

The DRLSE can be implemented with conventional finite different scheme. Due to the addition of the distance regularizing term, all spatial derivatives can be discretized using the more accurate and efficient central difference scheme (Li et al., 2010). The zero LS contour eventually approaches the grain boundary by minimizing the energy functional.

2.5.3 Numerical Demonstration

The DRLSE algorithm allows the use of more general functions as the initial LS functions. This is because the distance regularized internal energy functional $\mathcal{R}(\phi)$ could be considered as a diffusion term which adaptively increases or decreases $\nabla\phi$ and forces it to be close to one and therefore helps to maintain the signed distance property. Such diffusion is called a forward-and-backward (FAB) diffusion. This property comes in handy as it allows the initial guess of zero LS to be any reasonable function. For example, in Figure 2.11(b) and (c), the LS functions were initialized as binary step function (indicated as yellow squares), which takes constant negative values inside the yellow squares and positive values outside. In Figure 2.11(b), both grains were covered by one large initial LS function, while they are separately covered by different LS functions in (c). In both (b) and (c), the red lines showed how DRLSE

algorithm evolved as the result of minimizing the functional energy where the internal energy term restricts the LS function deviating from signed distance function, and external energy term provides specific image knowledge about where the grain boundaries are. Every step of this process substantially improved the results since functional term $\mathcal{A}(\phi)$, which shrinks or expands the LS function to minimize the enclosed weighted area, was a strong external force to drive the motion of contour. For image with weak object boundaries, this effect may cause boundary leak. Therefore, this term was removed in refinement steps, as indicated by blue lines.

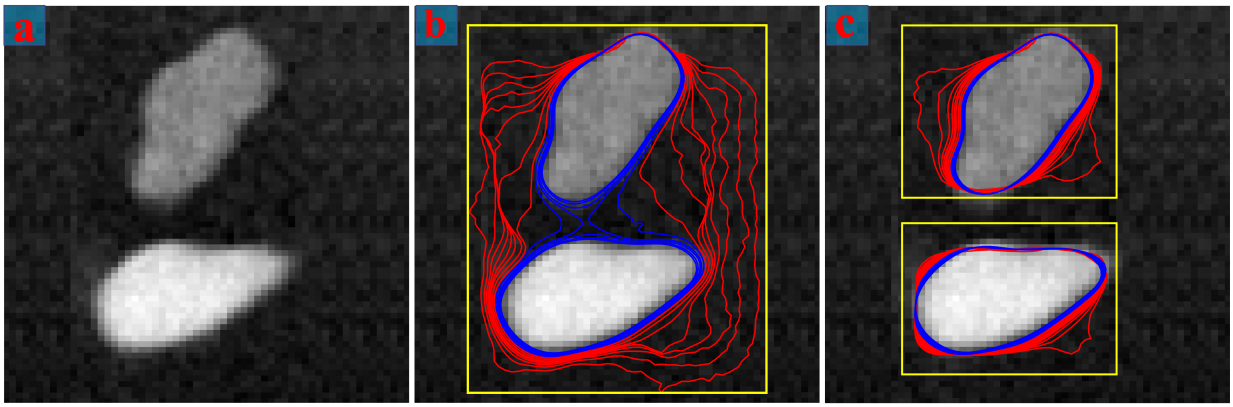

Figure 2.11: Evolution of LS algorithm. (a) Input image contains two grains, (b) Zero LS was initialized to cover both grains, (c) Zero LS was initialized for each grain. Yellow squares indicate initial guess of zero LS function, red contour lines indicate evolution of DRLSE algorithm and blue lines indicate the refinement steps.

Being able to accommodate any binary step function is computationally attractive, because it can take image segmentation results from a preliminary image binarization step. As the initial guess is likely to be already very close to the optimum result, it would take very few iterations to converge. Such initialization is desirable in some practical applications for its computational efficiency and simplicity. However, it does not perform well in our application because one XRCT image on granular material contains many grains. They sit close to each other, and the grain-grain or grain-void interfaces suffered severe partial volume effect. The DRLSE algorithm displays excellent capacity to consider topological changes, but it is often confused to correctly find boundaries in areas with multiple grains and it failed to disconnect two grains right next to each other. An example in shown in Figure 2.12. The DRLSE works well on images containing a single object or several well-separated objects. Therefore, before applying the DRLSE algorithm, the images are first binarized, passed through watershed algorithm to identify each individual grain, and separated grains into bounding boxes.

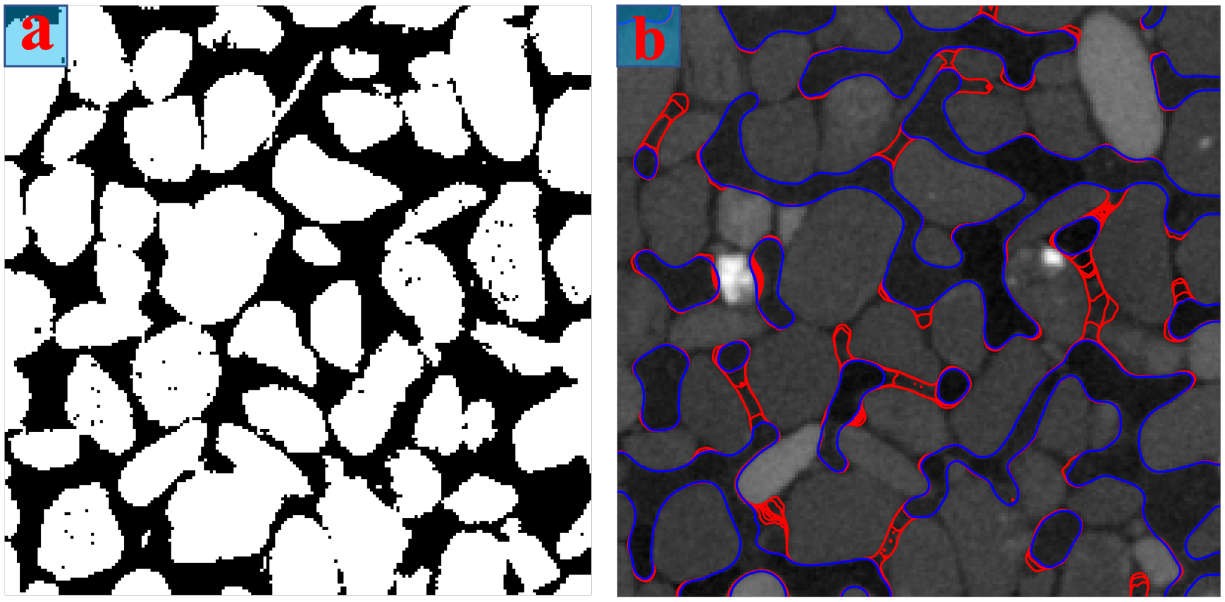

Figure 2.12: The DRLSE algorithm could not accurately converge to grain boundaries. (a) Binarized images used as binary step function to initialize LS function, (b) The evolution of zero LS contour lines in DRLSE algorithm.

If the binarization and watershed procedure fail to separate closely contacted grains, the DRLSE algorithm is not likely to separate them either. As a result, the attached clay adhesion and very fine grains are sometimes reconstructed as a part of the larger grains, hence they look like small bumps attached to the reconstructed avatars. This leads to abnormal grain shapes and inaccurate assembly fabric in the numerical modeling at the next stage. To remove those adhesion from the final reconstruction, the new functional term was introduced to run few more iterations after the DRLSE algorithm to penalize large surface curvature.

$$\mathcal{T}(\phi) = \int_{\Omega} k \|\nabla\phi\| d\Omega \tag{2.55}$$

$$k = \nabla \cdot \frac{\nabla\phi}{\|\nabla\phi\|} \tag{2.56}$$

This is inspired by the well-known mean curvature flow (Huisken, 1984) which originally proposed to describe behaviors of grain boundaries in annealing pure metal. Several reconstructed grains before and after introducing the curvature penalty term are illustrated in Figure 2.13, which substantially removes the small bumps but also produces a smoothed out grain surfaces. Therefore, a trade-off should be made between removing abnormal shapes and preserving surface details, and this is often made on a per-scan basis.

Figure 2.13: Comparison on the effect of introducing curvature penalty term.

2.6 Conclusion

We present a workflow for image preprocessing that includes image denoising, single image super resolution, and image segmentation. As a first step, the presented and analyzed NLM filter improved the quality of XRCT images of granular material by increasing the signal-to-noise ratio without impairing visible structures in the images. This highlights the NLM filter's utility in our application. Additionally, by using the root mean square error (RMSE) as an objective criterion to parameterize several denoised filters, the results indicated that the NLM filter outperformed more traditional local filters such as the Gaussian and Median filters in the XRCT denoising context. The primary disadvantage of the NLM filter is its computational cost, which is why we recommend using a GPU to process high-resolution images or large batches of images.

The following step is to propose an image super-resolution technique that is based on sparse signal representation and is capable of recovering high-resolution images from low-resolution input images. Mapping is accomplished by utilizing an overly-complete dictionary. We found that dictionaries of size 512 are sufficient to capture morphological components for XRCT images of granular material with repeated patterns in our numerical tests. The sparse representation also conveys robustness to noise, as the convex optimization formulation for learning dictionaries can be thought of as a MAP problem with the Laplacian prior as the sparsity requirement. Super-resolution images are advantageous for the subsequent stage of binarization because even advanced image binarization techniques did not work well on blurred low-resolution XRCT images. We require a super-resolution technique because our dataset was generated from a series of X-ray scans of a triaxial compression test, with only the beginning and ending scans being high-resolution. All other scans were taken while the experiment was temporarily paused, and thus were low-resolution to minimize experimental time.

Although the HMRF algorithm is not novel, this is the first attempt at application of a statistical model to binarize particle morphologies from XRCT images that incorporates

spatial information. Though this is the most critical step in identifying grains, there has been little advancement in the workflow's use of image segmentation in recent years. The HMRF algorithm strikes a balance between straightforward thresholding and more expensive neural networks. Even though the HMRF algorithm performed significantly better than other statistical models for high-resolution images in our application, it struggled with low-quality images. This is because the blurred images exhibit partial volume effect, particularly in the void pixels between two tightly packed grains. The HMRF algorithm is extremely robust, ensuring that the final segmentation remains stable even when the initial estimates are slightly different, and the intensity variability is high. This property makes the HMRF algorithm less sensitive to local intensity inhomogeneity and may result in incorrect results for poorly defined images. This issue can be resolved by weighting the void and grain phases differently, but this will only improve the results marginally because the assumption of MGM in pixel intensity distributions is inherently flawed. Indeed, because the EM algorithm is a local minimization method, it may become trapped in a local minimum and thus fail to find the appropriate thresholds when the intensity distribution is not Gaussian.

Finally, the grain boundaries from the segmented images are represented using the LS method. To simplify the numerical implementation, Distance Regularized LS Evolution (DRLSE) is used instead of the conventional signed distance function, which eliminates the need for regular LS function re-initialization. The DRLSE formulation is capable of maintaining the signed distance function's regularity intrinsically, which ensures accurate computation and stable LS evolution. By including a functional term to control how far the LS function deviates from the signed distance function, the algorithm becomes more robust and adaptive: the backward-and-forward diffusion property ensures that the zero LS remains at the boundaries. This is critical for grain fabric preservation and facilitates parameter tuning. The DRLSE algorithm is entirely based on edges and thus takes a very local approach to image segmentation. As such, prior to running the DRLSE algorithm, a pre-processing workflow is required to denoise, binarize, and label images. As illustrated in Figure 2.12, it was unable to segment multiple objects concurrently. This is also evidence that the DRLSE is flexible and efficient enough to generate signed distance function with a large number of reasonable initializations. Another penalty term is introduced into the evolution of the LS function, similar to the mean curvature flow, to address the issue of clay adhesion and much finer grains on the surface of the reconstructed avatars. However, as it tends to smooth out the whole grain surface, some morphological details are lost in the process.

Chapter 3

Parallel Implementation of LS-DEM with Hybrid MPI+OpenMP

3.1 Introduction

Grain morphology plays essential roles in determining the macroscopic properties of granular assemblies. Recent developments in the characterization of granular systems from X-ray computed tomographic (XRCT) images provide an excellent tool to digitalize and preserve soil fabric and grain morphologies, which enables the study of inter-grain interaction through numerical reconstruction of three-dimensional complex-shaped avatars from XRCT images. The grain-level morphological information can be integrated into a numerical method such as DEM to understand the link between granular material's macroscopic properties and its engineering behavior. Therefore, having a method that can model arbitrary grain shape is of paramount importance. However, large scale modeling requires significant computational cost and a DEM simulation on a single CPU machine might take several weeks to months to complete, making it infeasible for most applications. To alleviate the computational cost of DEM, grain shapes are often simplified and non-physical parameters such as rolling resistance are introduced in contact models. Existing DEM approaches account for grain morphology mostly by clustering or clumping spheres (Garcia et al., 2009; Tamadondar et al., 2019; Wu et al., 2021), using a simplex or polygon as a base geometry (Zhao et al., 2006; Zhao & Zhao, 2021) or generating realistic grains based on the concept of Fourier descriptors or spherical harmonic function (Garboczi, 2002; Taylor et al., 2006; Mollon & Zhao, 2012; Zhou et al., 2015). The first category is less appealing due to the lack of continuity in the curvatures and tangents. Other two are associated with high computational expenses with narrow-phase contact detection and force calculations. The idea of using LS to encode object shape, as proposed by Osher and Sethian (1988) has drawn substantial attention in the realm of image segmentation, as it is capable to fully capture the complex morphology of natural granular material with high-fidelity. The strength of LS-based algorithm is that they can track the motion of complex topology change on a fixed Eulerian grid. which is handy when dealing

with topological change as the curve evolves. The fundamental idea of LS based image segmentation is to implicitly represent boundaries of objects through grid-based interpolation from a space which is one dimension higher.

The concept of LS based morphological representation and DEM simulation are perfectly compatible with each other because of inter-grain interactions are straightforward in the in the LS framework, as shown by Vlahinić et al. (2013). The signed distance function is frequently used as the grid value in LS algorithms, with a positive grid value indicating a place outside the boundary and a negative grid value indicating a position inside the boundary. The extent to which a node penetrates a slave grain is interpolated from the associated LS function table in this sense. The direction of normal contact can be thought of as the gradient of the LS function at that node. In fact, grain morphology studies for large scale DEM simulation have received little attention even though the industrial importance is well known, the major reason is that large scale industrial discrete element simulations can often only afford to abstract grain shapes. In the modeling of arbitrarily complex grain shape, the problem size has impeded the general application of three-dimensional DEM for practical usage. To overcome these technical obstacles, it is desirable to optimize the code to perform the large-scale computational work using modern supercomputers. LS Discrete Element Method (LS-DEM) was introduced by Kawamoto (2018) in his PhD dissertation. Through high-fidelity LS reconstruction, Kawamoto (2018) investigated the kinematic and mechanical behavior of a system of discrete sand grains. In this section, we parallelized the three-dimensional LS-DEM algorithm for simulating complex-shaped granular grains.

Specifically, we parallelized a LS-DEM code with MPI using a variant of the binning algorithm and the spatial domain decomposition strategy to model arbitrarily complex-shaped grains with history-dependent contact model. Although our code is tailored to parallel the existing LS-DEM code, most parallel algorithms and implementation details can be migrated to applications in other disciplines. Many performance-critical implementation details are managed optimally to achieve high performance and scalability, such as locating a grain from their host bins in $O(1)$ complexity and vice versa, swapping ghost bins with an efficient message-passing algorithm, adapting a dynamic domain re-decomposition scheme, optimizing domain granularity for a problem given computing resources, collectively packing and transmitting migrated grains across adjacent MPI processors; implementing parallel algorithm for non-rigid type boundary, and reducing redundant contact resolution between grain pairs. The rest of this chapter is organized as follows, the concept of discrete element method (DEM) and the binning algorithm is first introduced; the contact model and numerical integration scheme used in LS-DEM (Kawamoto et al., 2016; Kawamoto, 2018) are described for completeness; then the MPI considerations and the recent implementation details are discussed. Finally, performance analyses, including speedup, efficiency, scalability, and granularity for different problem size (number of grains), of the modified code are presented.

3.2 Parallel Implementation of DEM

3.2.1 Contact Detection Complexity and Binning Algorithm

In DEM, each discrete element is considered as a separate body that interacts with neighboring elements. Therefore, conventional discrete element problem involves an enormous amount of contact detection, and an efficient solution of large-scale discrete element problems relies upon a fast and efficient contact detection algorithm should be employed for significant scale problem. In terms of the types of neighbor search algorithm, there are three typical neighbor search algorithms with various time complexities: $O(n^2)$, coming from a n-by-n complete mapping of an assembly of grains; $O(n \log n)$, multilevel grids rooted from a tree-based algorithm (Jagadish et al., 2005; Muja & Lowe, 2009); $O(n)$, the binning algorithm (Munjiza & Andrews, 1998; Williams et al., 2004) or the link cell algorithm (Grest et al., 1989). Yan and Regueiro (2018a) believed that the algorithms with three different complexities $O(n^2)$, $O(n \log n)$, and $O(n)$ only affect the performance of neighbor search and have no influences on the contact resolution if non-trivial shaped grains are studied, and the overall performance is highly limited for complex-shaped grains. Among these three neighbor search algorithms, many recent parallelism studies in DEM were based on the last kind; those algorithms share the same spirit but differ in terms of shape presentation, contact model, memory management, and performance optimization. The most straightforward and commonly used approach is the binning algorithm or the link-cell algorithm, which are essentially the same but slightly different in that geo-mechanics application only considers contact force. Three milestones in developing this binning algorithm are:

- Grest et al. (1989) introduced the layered link cell (LLC) approach for vectorizing molecular dynamics. This algorithm uses a Verlet table to maintain a neighbor list of all grains within a given cut-off distance of one another, periodically updates the table to take advantage of previous grain positions, employs a link cell algorithm to ensure that the CPU time per step is strictly proportional to the number of grains, and takes advantage of Newton's third law to resolve contact with only half of the adjoining cells.

- Munjiza and Andrews (1998) described a no binary search (NBS) algorithm for encoding element location using a linked list that works equally well for dense and loose packing, with CPU time remaining constant and RAM requirements growing insignificantly as packing density decreases. The NBS contact detection technique is based on the assumption that each discrete element is approximated by an identical circular discs and that the computational domain is bounded by a box.

- Williams et al. (2004) proposed the CGRID algorithm, a generalized binning technique that extends the NBS approach to handle assemblies of any shape or size. Their contact detection approach partitions a D-dimensional space recursively into bins containing items that are currently defined in D-1 dimensional space. To accommodate grains of varying sizes, this algorithm chooses the bin size irrespective of the grain

sizes and allows for the extension of large grains across multiple bins. While this approach is efficient and resilient when implemented sequentially, it may not be suited for parallelization.

3.2.2 Domain Decomposition Strategy

The idea of the binning algorithms is to place each grain into a bin using a hash on the grain's coordinates. Once the grains are sorted into bins, one can reason about the spatial closeness based solely on the bins' fixed relationships. The bins are created through a domain decomposition strategy: the computational domain is first divided into sub-domains. Each MPI processor carries out calculations on its respective portions of the data domain. Then every sub-domain is further partitioned into many bins to hold the objects of interest, grains in the case considered herein. The size of bin is larger than the largest diameter of an assembly of grains such that any pair of grains that are separated by more than one bins are not considered to be in contact. An illustration of domain decomposition strategy can be found in Figure 3.1 which also denotes the notions of bin, block, and border layer. The main advantage of using the domain decomposition strategy is high scalability even for a large number of processors, and its usability for both shared and distributed architecture machines Gopalakrishnan and Tafti (2013). Notably, the domain decomposition strategy is not limited to discrete element modeling, Zohdi and Wriggers (1999) developed a similar technique for reducing the computational complexity of boundary value problems associated with structural analysis of bodies with arbitrary external geometry and heterogeneous microstructure. They partition and decouple the heterogeneous body into more computationally tractable, non-overlapping subdomains and the subdomain boundary conditions is approximated. There has been considerable interest in developing parallel DEM codes in recent years (Henty, 2000; Baugh Jr & Konduri, 2001; Washington & Meegoda, 2003; Maknickas et al., 2006; Walther & Sbalzarini, 2009; Chorley & Walker, 2010; Kačianauskas et al., 2010; Gopalakrishnan & Tafti, 2013; Amritkar et al., 2014), a comprehensive review is given in Yan and Regueiro (2018b) and Yan and Regueiro (2019).

In general, most parallel implementations of DEM only deal with spheres rather than complex-shaped grains except Yan and Regueiro (2018b). Although it is acceptable to sacrifice grain size and shape in the trade-off larger problem size in the prototype-scale simulations for engineering studies, overly simplified grains are insufficient to develop generalizations about fundamental micromechanics without capturing grain-level details as pointed out by Peters et al. (2009). Usually, modeling arbitrarily shaped particles demands several orders of magnitude more computational effort than a spherical DEM. For example, the CPU demand for simulating 122 million spheres is equivalent to simulating 488k poly-ellipsoids when using the same inter-grain contact model Yan and Regueiro (2018b). This is also a major computational bottleneck in LS-DEM. The geometric basis of a grain is embedded in a LS table and discretized into hundreds of nodes as seeded on the surface. Each node on the surface of the particle is checked for contact in the force resolution step, which results in several order higher computational cost than for spherical grains. Another complexity arises when con-

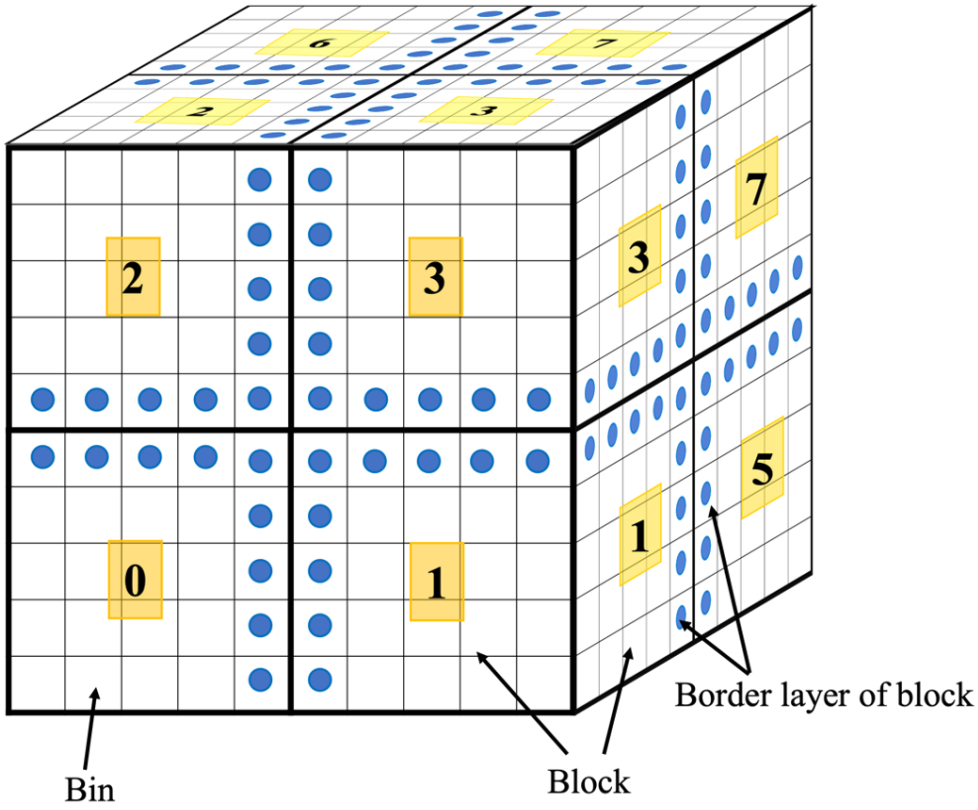

Figure 3.1: Illustration of domain decomposition strategy, it contains eight sub-domains, blue dots are boundary bins that require communication between processors.

sidering history-dependent tangential behavior for granular materials. Modeling arbitrarily complex grains and a more sophisticated inter-grain contact model is not difficult in a sequential code. However, it becomes a lot more cumbersome when the code is parallelized, as tracking the loading-unloading-reloading path and contact histories is not straightforward. Consequently, the domain decomposition parallel algorithm has to be modified for different inter-grain contact models, grain shape complexity, boundary conditions, and computational granularity.

3.3 Introduction to DEM and parallel LS-DEM implementation

A completed LS-DEM code is made up of several critical components, including grain geometry representation, inter-grain contact models, contact detection and force resolution, time integration scheme, damping mechanism, boundary conditions, and modeling of various

loading conditions (Yan & Regueiro, 2018b). For the sake of completeness, these components of the code are briefly presented herein, and additional details are found in Kawamoto (2018), Yan and Regueiro (2018b) and Bandeira and Zohdi (2019).

3.3.1 Contact Detection and Resolution Algorithm

The LS reconstructed avatars are represented with hundreds of discretized nodes, and the volume, mass and moments of inertia are computed by counting the number of voxels inside avatars. In terms of computational cost, the signed distance function can be calculated efficiently using a marching method (Sethian, 1996). This is a one-time cost in constructing a rigid body model. For efficiency, the object is stored with the center of mass at the origin and the axes aligned with the principal axes of inertia resulting in a diagonal inertia tensor to simplify many calculations. The interferences between two implicit surfaces are found by checking the sign of nodes in the signed-distance function of the other. This is not sufficient to detect all collisions, as edge-face collisions are missed when both edge vertices are outside the implicit surface. Since the errors are proportional to the edge length, but they can be ignored in a well resolved mesh with sufficient node density.

The particle overlap model is the most commonly used model which determines the contact force from the separation distance between particles and material properties, the assumption underlain this model is that particles are assumed to be spheres (Zohdi, 2017). Typically, the contact detection and resolution algorithm phases are the primary computational bottlenecks, especially when simulating complex-shaped grains with hundreds of nodes per grain. If nonlinear history-dependent mechanical models are used, this stage becomes even more computationally demanding. The contact identification and resolution algorithm phases in most DEM codes programs are divided into two sub-phases: nearest neighbor search (or spatial resolution) and contact resolution. A neighbor search phase identifies or estimates objects that are close to the target object using an easy-to-model approximate geometry, such as a bounding box or a bounding sphere. The LS-DEM code adjusts to the following:

$$\|\mathbf{c}^i - \mathbf{c}^j\| > r^i + r^j \tag{3.1}$$

Where $\mathbf{c}^i$ is the position of the mass center of grain i and r^i is the equivalent radius of grain i. Following that, the contact resolution phase employs a geometric representation of each body. At the resolution phase, the LS reconstructed avatars comprise hundreds of discretized nodes, each of which must be checked to the surface of a neighboring avatar. This stage is performed sequentially and has a minimal vectorization potential due to the explicit nature of DEM modeling. This is the cost of a three-dimensional DEM code capable of simulating grains of any shape. In this aspect, resolving the contact between two arbitrarily shaped grains is significantly more computationally expensive than resolving the contact between two grains having basic geometrical representations, such as spheres. Due to the necessity for numerical precision and resilience, it frequently increases the floating-point operations by many orders of magnitude. We used the binning algorithm to limit the scope of the

neighbor search; the binning algorithm's fundamental idea is to improve spatial locality, which means that a discrete grain does not need to check with all other grains but only those in its immediate vicinity. However, the total computing benefit of neighbor search may be negligible for complex-shaped grains, as neighbor search accounts for a small fraction of floating-point operations. As a result, the most computationally intensive portion of the LS-DEM is the force resolution phase, rather than the neighbor detection phase which is the major bottleneck for conventional DEM simulating disks or spheres. We later show that the idea of binning algorithm is well compatible with domain decomposition since many parallel implementations consider bin as a base unit.

3.3.2 Search Complexity of the Binning Algorithm

The binning method assumes that grain interactions occur only when two grains come into contact. The node-to-surface contact is solved explicitly and is not vectorized. The computational cost of an n-by-n naive search algorithm is well-known to be $O(n^2)$ for a generalized N-body problem (Gray & Moore, 2000). Supposed the non-contact pairs of objects are excluded, the computational complexity is reduced from $O(n^2)$ to $O(n)$ for the binning algorithm or to $O(n \log n)$ for a tree algorithm. In this particular problem, the idea of binning is to place each grain into a bin of prescribed size hashed on the grain's coordinates. The cutoff distance for DEM is chosen to be at least the largest equivalent diameter of grains so that any two grains which are at least one bin distance away will not interact. Once the grains are sorted into bins, one can evaluate the spatial proximity based solely on the fixed relationships of the bins. As shown in Figure 3.2, the grain filled in yellow and marked in orange line represent the one of interest, and blue grains represent those that require check for detection. As can be seen, the computational effort for neighbor detection is substantially decreased through binning especially for spherical particles (Zohdi, 2004a; Zohdi, 2010; Zohdi, 2012). Due to the 3D shape of grains, i.e., grains cannot be clustered arbitrarily dense, we can assume the average number of grains in each bin is b, then the computational complexity is reduced to $O(n)$ with a small constant b. However, it can be shown that the $O(n)$ neighbor search algorithm might be inferior to $O(n^2)$ for computing a very small number of grains for two reasons. 1) Both shared-memory and distributed-memory encounter synchronization and communication overheads, which are inevitable and always retards the performance, primarily when the performance is governed by communication bandwidth and latency. 2) The overall performance improvement resulting from a parallel algorithm might be highly limited for complex-shaped grains, because the algorithm only affects the performance of the neighbor estimate rather than the contact resolution. The contact resolution step takes up a large fraction of floating-point operations in the computation. For these reasons, it is acceptable and advisable to use the serial approach when the number of grains is relatively small. Of course, serial implementation becomes extremely inefficient as the number of grains increases.

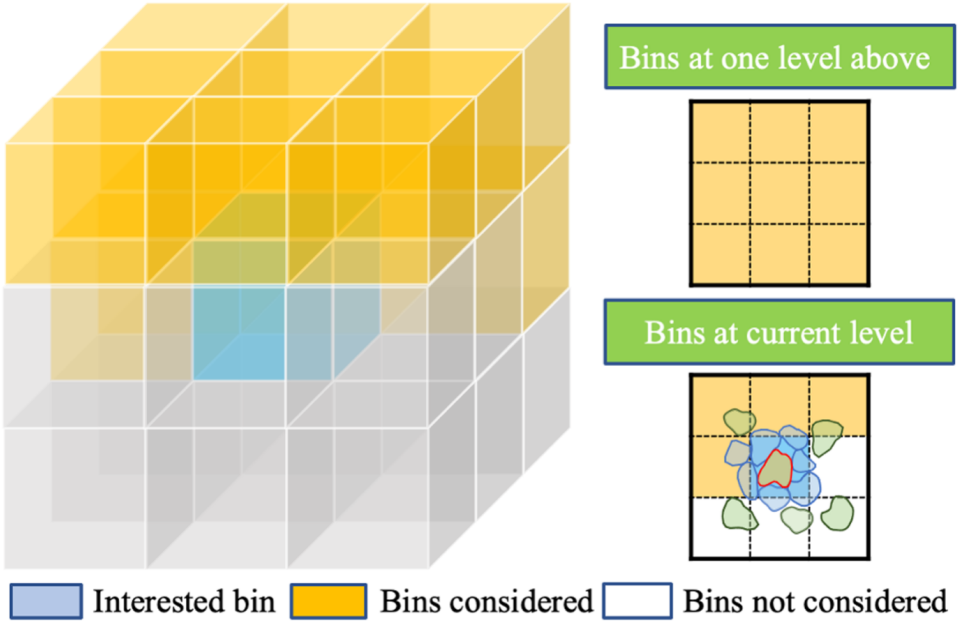

Figure 3.2: Illustration of neighbor search in binning algorithm and Computational efforts reduced by half due to symmetry

3.3.3 Inter-Grain Normal Contact Forces

The LS based shape representation and contact algorithms unique to LS-DEM were developed by Kawamoto et al. (2016) and are followed herein. Contact in LS-DEM is handled through iterating node-to-surface contact algorithm, whereby nodes are seeded onto the surface of each grain and grain shape is implicitly embedded in its LS grid value table.

The discretized nodes that characterize the grains participate in the actual interaction computations. This representation is similar to a triangulated surface mesh except that LS-DEM does not store connectivity information between nodes and therefore does not consider edge-surface collision. The density of nodes on a given grain is a matter of choice and is entirely up to the designer. The number of nodes seeded onto a grain has no effect on the underlying geometry but does have an effect on the computational complexity associated with force resolution. With grain refining, extremely high-fidelity reconstruction frequently requires unaffordable computation time. Lim et al. (2014) demonstrate that seeding with a maximum node-to-node spacing less than $d/10$, where d is the grain diameter, is sufficient to capture grain morphology and a further increase in nodal densities has a minor impact on behavior.

Contact is determined by comparing each node of a master grain to a slave grain for penetration, and the computational cost of contact resolution is proportional to the number of nodes seeded onto the master grain. By embedding the grain in a three-dimensional Cartesian grid with a value indicating the signed distance to the nearest grain surface, the

grain surface is implicitly defined by a set of nodes with zero LS value. This framework is quite convenient as it is amenable to calculating the forces between grains with the commonly used penalty-based method. As shown in Figure 3.3, the two quantities of most interests are the LS value $\phi(\mathbf{p})$ and its gradient $\nabla\phi(\mathbf{p})$ for any point $\mathbf{p}$ in the Cartesian grid as they are exactly the amount of penetration and contact normal direction. They can be computed through interpolation from grid values near $\mathbf{p}$, any order of interpolation can be used, and linear interpolation was used here for simplicity and speed.

where $d_k^{j,i}$ denotes the scalar penetration of the k-th node on grain i to the geometry of grain j; ϕ^j is the LS function of grain j; $\mathbf{p}_k^j$ is the position of k-th node on grain i considered in grain j's coordinate; $\hat{\mathbf{n}}_k^{j,i}$ is the unit normal direction of penetration $d_k^{j,i}$. Note that one property of LS accommodated with signed distance function is that the gradient of a point of LS is unit at that point. However, due to the LS function's discrete nature, the magnitude of $\nabla\phi^j(\mathbf{p}_k^j)$ is very close but not equal to unity and therefore it is normalized. If at least one node $\mathbf{p}_k^j$ of master grain i is penetrating a slave grain j, then the two grains are considered to be in contact and inter-grain forces are computed. This process is detailed in Algorithm 1:

The proposed code adopts the linear elastic contact model. Thus, the normal contact force contributed from the node $\mathbf{p}_k^i$ on grain i is:

$$\mathbf{F}_{n,k}^i = \begin{cases} -k_n d_k^{j,i} \hat{\mathbf{n}}_k^{j,i} & d_k^{j,i} < 0 \\ 0 \quad \text{else} \end{cases} \tag{3.2}$$

Where K_n is the normal contact stiffness. By action and reaction, the contribution of contact normal force $\mathbf{F}_{n,k}^j$ from the node $\mathbf{p}_k^i$ on grain j is:

$$\mathbf{F}_{n,k}^j = -\mathbf{F}_{n,k}^i \tag{3.3}$$

The moment $\mathbf{M}_{n,k}^i$ contributed by the normal contact force $\mathbf{F}_{n,k}^i$ at the node $\mathbf{p}_k^i$ on grain i is:

$$\mathbf{M}_{n,k}^i = (\mathbf{p}_k^i - \mathbf{c}^i) \times \mathbf{F}_{n,k}^i \tag{3.4}$$

Where $\mathbf{c}^i$ is the centroid of grain i. Similarly, the moment $\mathbf{M}_{n,k}^j$ contributed by the normal contact force $\mathbf{F}_{n,k}^j$ at the node $\mathbf{p}_k^i$ on grain j is:

$$\mathbf{M}_{n,k}^j = (\mathbf{p}_k^i - \mathbf{c}^j) \times \mathbf{F}_{n,k}^j \tag{3.5}$$

It is critical to keep in mind that the contact forces between two grains vary slightly depending on which grain is selected as the master grain. This is because the k-th node on master grain i might penetrate the slave grain j, while there does not exit a corresponding node on slave grain j penetrating master grain i due to the discrete nature of LS geometry representation. This does not influence the serial implementation as the force resolution phase is always iterated from a small index to a large index. Nevertheless, the index order

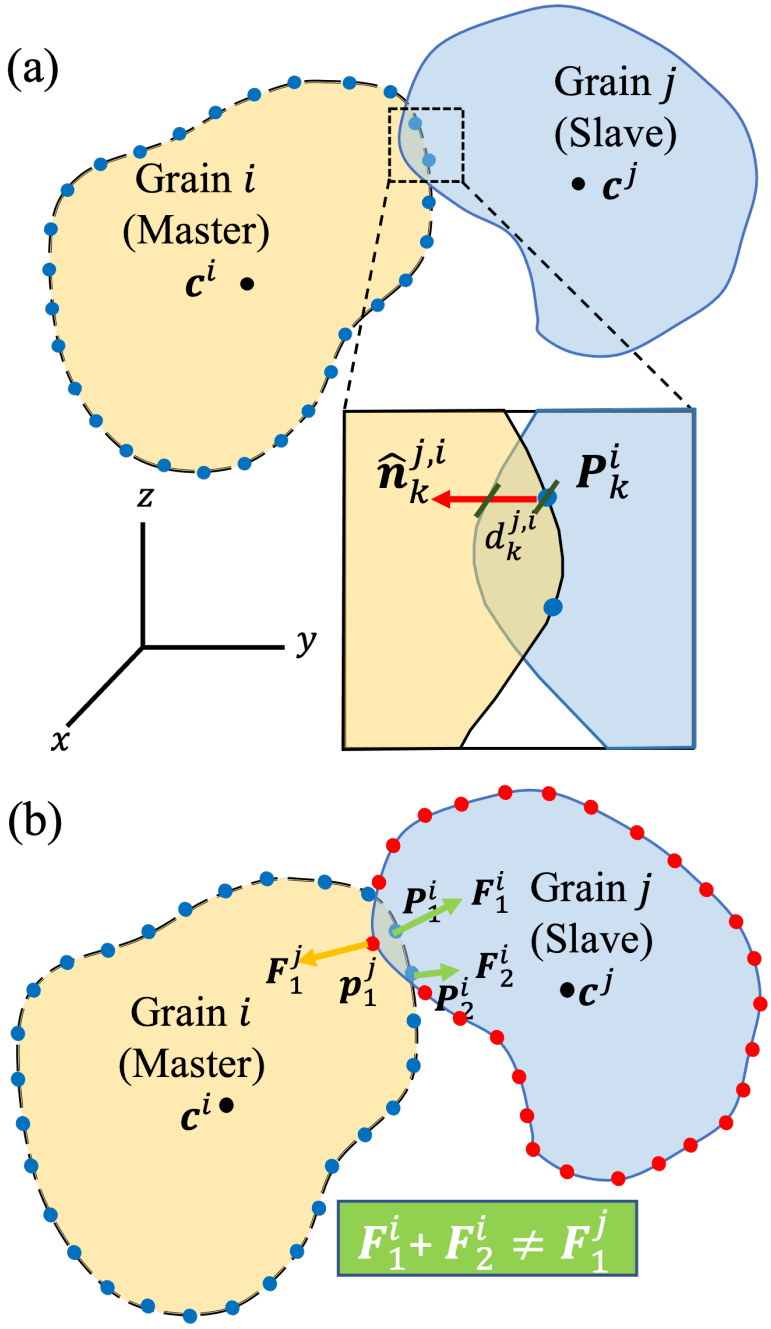

Figure 3.3: (a) Illustration of two contact grains, where $d_k^{j,i}$ denotes the scalar penetration of the k-th node on grain i to the geometry of grain j, $\hat{\mathbf{n}}_k^{j,i}$ is the unit normal of penetration $d_k^{j,i}$. (b) Contact forces between two grains are different.

Algorithm 1 findPenetrationDirection

INPUT: grain, point $\mathbf{P}$
OUTPUT: flag, penetration depth d, contact normal in principal frame $\tilde{\mathbf{n}}$
/* extract LS values nearby $\mathbf{P}$ */
/* $\mathbf{P}_x$, $\mathbf{P}_y$, $\mathbf{P}_z$ are coordinates of $\mathbf{P}$ */
$x_0 = \mathbf{floor}(\mathbf{P}_x)$, $y_0 = \mathbf{floor}(\mathbf{P}_y)$, $z_0 = \mathbf{floor}(\mathbf{P}_z)$
$x_1 = \mathbf{ceil}(\mathbf{P}_x)$, $y_1 = \mathbf{ceil}(\mathbf{P}_y)$, $z_1 = \mathbf{ceil}(\mathbf{P}_z)$
/* function **getGridValue** looks up the LS table to extract value */
$P_{000} = \mathbf{getGridValue}(x_0, y_0, z_0)$
$P_{001} = \mathbf{getGridValue}(x_0, y_0, z_1)$
$P_{010} = \mathbf{getGridValue}(x_0, y_1, z_0)$
$P_{011} = \mathbf{getGridValue}(x_0, y_1, z_1)$
$P_{101} = \mathbf{getGridValue}(x_1, y_0, z_1)$
$P_{100} = \mathbf{getGridValue}(x_1, y_0, z_0)$
$P_{110} = \mathbf{getGridValue}(x_1, y_1, z_0)$
$P_{111} = \mathbf{getGridValue}(x_1, y_1, z_1)$
/* find penetration d via linear interpolation */
$P_x = P_{100} - P_{000}$
$P_y = P_{010} - P_{000}$
$P_z = P_{001} - P_{000}$
$P_{xy} = -P_x - P_{010} + P_{110}$
$P_{xz} = -P_x - P_{001} + P_{101}$
$P_{yz} = -P_y - P_{001} + P_{011}$
$P_{xyz} = P_{xy} - P_{001} - P_{101} - P_{011} + P_{111}$
$\Delta x = \mathbf{P}_x - x_0$
$\Delta y = \mathbf{P}_y - y_0$
$\Delta z = \mathbf{P}_z - z_0$
$d = P_{000} + P_x \cdot \Delta x + P_y \cdot \Delta y + P_z \cdot \Delta z + P_{xy} \cdot \Delta x \cdot \Delta y + P_{xz} \cdot \Delta x \cdot \Delta z + P_{yz} \cdot \Delta y \cdot \Delta z + P_{xyz} \cdot \Delta x \cdot \Delta y \cdot \Delta z$
if $d < 0$ **then**
    flag = **True**
/* take derivative respect to d obtain contact normal $\tilde{\mathbf{n}}$ */
$\tilde{\mathbf{n}} = P_x + P_{xy} \cdot \Delta y + P_{xz} \cdot \Delta z + P_{yz} \cdot \Delta y \cdot \Delta z + P_{xyz} \cdot \Delta y \cdot \Delta z$
return d, $\tilde{\mathbf{n}}$

will change after adding or deleting migrated grains from bins, thus we always consider the grain with the smaller index as the master grain in force resolution in the parallel implementation.

To maximize efficiency, the grain is stored with the center of mass at the origin and the axes aligned with the inertia primary axes, resulting in a diagonal inertia tensor that simplifies many calculations. As a result of the rigid body assumption, grain's LS function is never altered. When contact is computed, the nodes $\mathbf{p}_k^i$ of grain i are temporarily relocated

into the reference configuration of grain j's LS function. The contact forces and moments are then determined (in the reference configuration of grain j) and translated back to the global frame.

3.3.4 Inter-Grain Tangential Contact Forces

The normal contact force is directly dependent on the extent to which a seeded node on the master grain overlaps with the geometry of the slave grain. Due to the fact that penetration can be calculated directly in the LS formulation, the normal contact force is not history dependent. In comparison, the contact tangential force can be either history-dependent or independent of history. A straightforward illustration of a history-independent tangential contact model is one in which the tangential force is always proportional to its normal equivalents. The original LS-DEM code (Kawamoto et al., 2016) considers a history-dependent Coulomb friction model, which requires that contact histories accompany migrating grains. This is because the history-dependent approach calculates tangential displacement increments rather than accumulated tangential displacement until two objects are separated. It is vital to determine if a code uses a history-dependent contact model or not, as transferring shear history data is a necessary but tedious procedure due to the large and varied number of nodes seeded on each grain. In the case of a history-dependent contact model, grains must be considered individually for cross-block migration. Typically, a history-dependent tangential model is required to put the simulation on a par with physical experiments, since highly simplified inter-grain contact models are incapable of accurately capturing the physical properties of frictional granular material because they do not account for shear history and do not simulate non-linearity. While the Coulomb friction model is the simplest, more sophisticated models incorporate the rate of shearing, and incorporating such models may result in improved results.

To compute the frictional forces, LS-DEM uses a Coulomb friction model similar to those in Cundall and Strack (1979). For a given node $\mathbf{p}_k^i$, frictional forces and the related moments only exist if $\mathbf{F}_{n,k}^i \neq 0$. The relative velocity $\mathbf{v}_k$ of node $\mathbf{p}_k^i$ to grain j is:

$$\mathbf{v}_k = \mathbf{v}^i + \boldsymbol{\omega}^i \times (\mathbf{p}_k^i - \mathbf{c}^i) - \mathbf{v}^i - \boldsymbol{\omega}^j \times (\mathbf{p}_k^i - \mathbf{c}^j) \tag{3.6}$$

Where $\mathbf{v}^i$, $\mathbf{v}^j$, $\boldsymbol{\omega}^i$, $\boldsymbol{\omega}^j$ are translational and angular velocities of grain i and grain j. The incremental shear displacement $\Delta\mathbf{s}_k$ is then:

$$\Delta\mathbf{s}_k = [\mathbf{v}_k - (\mathbf{v}_k \cdot \hat{\mathbf{n}}_k^{j,i})\hat{\mathbf{n}}_k^{j,i}] \cdot \Delta t \tag{3.7}$$

The shear force $\mathbf{F}_{s,k}^i$ on grain i contributed by node $\mathbf{p}_k^i$ is updated as such:

$$\mathbf{F}_{s,k}^i = \mathbf{Z}\mathbf{F}_{s,k}^i - k_s \Delta\mathbf{s}_k \tag{3.8}$$

Where $\mathbf{Z}$ is the rotation operation that rotates the normal vector $\hat{\mathbf{n}}_k^{j,i}$ at the previous timestep to the normal vector at the current timestep and k_s is the shear contact stiffness.

This step is necessary because the relative orientation of the two grains would change between timesteps. In the code presented herein, we use Rodrigues's rotation formula (Murray et al., 1994):

$$\mathbf{v}_{rot} = \cos\theta + (1 - \cos\theta)(\mathbf{k} \cdot \mathbf{v})\mathbf{k} + \sin\theta \mathbf{k} \times \mathbf{v} \tag{3.9}$$

Where θ is the angle between the interested vector in two timesteps, and $\mathbf{k}$ is cross product between normal vector at the current and previous timestep. The Coulomb friction law dictates $\mathbf{F}^i_{s,k}$ be capped at a fraction of the normal force $\mathbf{F}^i_{n,k}$:

$$\mathbf{F}^i_{s,k} = \frac{\mathbf{F}^i_{s,k}}{\|\mathbf{F}^i_{s,k}\|} \min\left(\|\mathbf{F}^i_{s,k}\|, \mu\|\mathbf{F}^i_{s,n}\|\right) \tag{3.10}$$

Where μ is the inter-grain friction coefficient. By action and reaction:

$$\mathbf{F}^j_{s,k} = -\mathbf{F}^i_{s,k} \tag{3.11}$$

The moment $\mathbf{M}^i_{s,k}$ contributed by node $\mathbf{p}^i_k$'s shear force on grain i is:

$$\mathbf{M}^i_{s,k} = (\mathbf{m}^i_k - \mathbf{c}^i) \times \mathbf{F}^i_{s,k} \tag{3.12}$$

Similarly, the $\mathbf{M}^j_{s,k}$ contributed by node $\mathbf{p}^i_k$'s shear force on grain j is:

$$\mathbf{M}^j_{s,k} = (\mathbf{m}^i_k - \mathbf{c}^j) \times \mathbf{F}^j_{s,k} \tag{3.13}$$

In the end, the total contact force on grain i is found by summing all nodal contact forces:

$$\mathbf{F}^i_{rot} = \sum_{k=1}^{N} (\mathbf{F}^i_{n,k} + \mathbf{F}^i_{s,k}) \tag{3.14}$$

Where N is the number of nodes on grain i. By action and reaction:

$$\mathbf{F}^j_{rot} = -\mathbf{F}^i_{rot} \tag{3.15}$$

The total contact moment on each grain is found by summing all nodal contact moments:

$$\begin{aligned} \mathbf{M}^i_{rot} &= \sum_{k=1}^{N} (\mathbf{M}^i_{n,k} + \mathbf{M}^i_{s,k}) \\ \mathbf{M}^j_{rot} &= \sum_{k=1}^{N} (\mathbf{M}^j_{n,k} + \mathbf{M}^j_{s,k}) \end{aligned} \tag{3.16}$$

To conclude, the inter-grain interaction is shown in Algorithm 2 below.

Algorithm 2 findInterGrainForceMoment

INPUT:grain A, grain B
OUTPUT: grainForce $\mathbf{F}$, grainMoment $\mathbf{M}$
for int $i = 0$; $i <$ **num_nodes**; $i = i + 1$ **do**
  $\mathbf{u^{AB}} = \mathbf{u_A} - \mathbf{u_B}$, where $\mathbf{u_A}$ is the mass center of grain A
  if $|\mathbf{u^{AB}}| < r_A$, $r_A = \max_{k\in\mathcal{N}}\{|\mathbf{u_A^k}|\}$, where $\mathbf{u_A^k}$ is the vector from k-th node of grain A to mass center **then**
    /* rotate $\mathbf{u_A^i}$ to the principal frame of grain B's LS grid */
    $\tilde{\mathbf{u}}_A^i = \mathbf{R}^T\mathbf{u_A^i} + \mathbf{c_B}$, where R is the operator rotate vector from principal frame to global frame
    /* check if $\tilde{\mathbf{u}}_A^i$ penetrates into grain B */
    flag, d^i, $\tilde{\mathbf{n}}^i$ = **findPenetration**($\tilde{\mathbf{u}}_A^i$, grain B)
    if flag **then**
      /* rotate contact normal $\tilde{\mathbf{n}}^i$ to global frame */
      $\mathbf{n^i} = \mathbf{R}\tilde{\mathbf{n}}^i$
      /* Compute spring force at normal direction */
      $\mathbf{f_n^i} = k_n \cdot d^i \cdot \mathbf{n^i}$
      $\mathbf{F} = \mathbf{F} + \mathbf{f_n^i}, \quad \mathbf{M} = \mathbf{M} + \mathbf{u_A^i} \times \mathbf{f_n^i}$
      /* Compute relative velocity of two grains at $\mathbf{u_A^i}$ */
      $\mathbf{v_{AB}^i} = \mathbf{v_A} - \mathbf{v_B} + \omega_\mathbf{A}^\mathbf{i}\mathbf{u_A^i} - \omega_\mathbf{B}^\mathbf{i}\mathbf{u_B^i}$
      /* Compute friction increment in tangential direction */
      $\mathbf{f_t^{t+\Delta t,i}} = \mathbf{Z}\mathbf{f_t^{t,i}} - \mathbf{v_{AB}^i} \cdot k_t \cdot \Delta t$, where $\mathbf{Z}$ is operator rotate past shear force direction to current direction.
      /* Check Coulomb's friction criterion is satisfied */
      $f^i = \min\{|\mathbf{f_t^i}|, \mu|\mathbf{f_n^i}|\}$
      $\mathbf{f_t^i} := f^i \cdot \mathbf{f_t^i}/|\mathbf{f_t^i}|$
      $\mathbf{F} = \mathbf{F} + \mathbf{f_t^i}, \quad \mathbf{M} = \mathbf{M} + \mathbf{u_A^i} \times \mathbf{f_t^i}$
  return $\mathbf{F}$, $\mathbf{M}$

3.3.5 Discrete Equations of Motion

The scheme described here was implemented by Kawamoto et al. (2016), who adopted the work by Lim and Andrade (2014) to update the center of mass and nodes of each grain. This scheme solves Newton's and Euler's governing equations of motion. At the end of each timestep, the locations, forces, and velocities of the grains are known, allowing the grain motion to be explicitly updated via Newton's law's governing translational equation:

$$ma_i + Cv_i = F_i \tag{3.17}$$

Where $i = 1, 2, 3$ in three dimensions, m is the mass of the grain, $C = \xi m$ is the damping that proportionally scales the linear velocity v_i, with ξ being the global damping parameter. The linear acceleration is given by a_i and is related to the resultant force F_i. To integrate

the translational components of motion, the centered finite-difference integration scheme is used.

$$v_i^{n+\frac{1}{2}} = \frac{1}{1+\frac{\xi\Delta t}{2}}[(1-\frac{\xi\Delta t}{2})v_i^{n-\frac{1}{2}} + \frac{\Delta t}{m}F_i] \tag{3.18}$$

$$x_i^{n+1} = x_i^n + \Delta t v_i^{n+\frac{1}{2}} \tag{3.19}$$

This scheme is second-order explicit which is conditionally stable, and we admit that there is a family trapezoidal integration schemes in various forms. Based on the variable metric, the scheme becomes implicit, unconditionally stable and a coupled system of equations, which are solved using an adaptive iterative scheme. (Zohdi, 2003; Zohdi, 2004b; Zohdi, 2007; Zohdi, 2013). For complex-shaped object, the rotational components of motion should also be integrated, the time derivatives of the angular accelerations in the principal frame are given by Euler's equations of motion.

$$\dot{\boldsymbol{\omega}} = (M - \boldsymbol{\omega} \times (I\boldsymbol{\omega}) - \xi I\boldsymbol{\omega})/I \tag{3.20}$$

Where $\dot{\boldsymbol{\omega}}$ is the angular acceleration, $\boldsymbol{\omega}$ is the angular velocity, I is the (diagonal) moment of inertial tensor in the principal body-fixed frame, and M is the torque vector in the principal body-fixed frame. The Euler equations are nonlinear due to the presence of angular velocities products on both side after numerically discretizing above formulas. Therefore, to appropriately integrate the rotational components of motion, a predictor-corrector procedure is recommended:

(1) Estimate the angular velocities at the current timestep by assuming constant angular acceleration for an additional half step.

$$\omega_i^{'n} = \omega_i^{n-\frac{1}{2}} + \frac{1}{2}\Delta\omega_i^{n-1} \tag{3.21}$$

where $\Delta\omega_i^{n-1} = \alpha_i^{n-1}\Delta t$

(2) Calculate angular velocity predictor by using the estimates as mentioned earlier.

$$\begin{aligned}
\Delta\omega_1^{'n} &= \Delta t[M_1^n + \omega_2^{'n}\omega_3^{'n}(I_2 - I_3) - \xi I_1\omega_1^{'n}]/I_1 \\
\Delta\omega_2^{'n} &= \Delta t[M_2^n + \omega_3^{'n}\omega_1^{'n}(I_3 - I_1) - \xi I_2\omega_2^{'n}]/I_2 \\
\Delta\omega_3^{'n} &= \Delta t[M_3^n + \omega_1^{'n}\omega_2^{'n}(I_1 - I_2) - \xi I_3\omega_3^{'n}]/I_3
\end{aligned} \tag{3.22}$$

(3) Predict angular velocities at the current timestep.

$$\omega_i^n = \omega_i^{n-\frac{1}{2}} + \frac{1}{2}\Delta\omega_i^{'n} \tag{3.23}$$

(4) Calculate angular velocity correctors.

$$\begin{aligned}\Delta\omega_1^n &= \Delta t[M_1^n + \omega_2^n\omega_3^n(I_2 - I_3) - \xi I_1\omega_1^n]/I_1 \\ \Delta\omega_2^n &= \Delta t[M_2^n + \omega_3^n\omega_1^n(I_3 - I_1) - \xi I_2\omega_2^n]/I_2 \\ \Delta\omega_3^n &= \Delta t[M_3^n + \omega_1^n\omega_2^n(I_1 - I_2) - \xi I_3\omega_3^n]/I_3\end{aligned} \tag{3.24}$$

(5) Update angular velocities by using the correctors.

$$\omega_i^{n+\frac{1}{2}} = \omega_i^{n-\frac{1}{2}} + \frac{1}{2}\Delta\omega_i^n \tag{3.25}$$

For small time steps used to resolve the inter-grain contacts and for quasi-static conditions in which the angular velocities are small, the number of iterations is typically small. Usually, between three and five iterations are required to achieve machine precision tolerance. Orientations for each grain are updated using Evans' singularity free quaternion approach (Evans & Murad, 1977). For Euler's equations of motion and the integration of the quaternions, the torques are specified in the body or principal frame, while the contact detection and force calculations are performed in a space or global frame. Therefore, the rotation matrix from space to body frame is given by:

$$R = \begin{pmatrix} -q_1^2 + q_2^2 - q_3^2 + q_4^2 & -2(q_1q_2 - q_3q_4) & 2(q_2q_3 + q_1q_4) \\ -2(q_1q_2 + q_3q_4) & q_1^2 - q_2^2 - q_3^2 + q_4^2 & -2(q_1q_3 - q_2q_4) \\ 2(q_2q_3 - q_1q_4) & -2(q_1q_3 + q_2q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{pmatrix} \tag{3.26}$$

Where the q's are the quaternions of Evans and Murad (1977).

$$\begin{aligned} q_1 &= \sin\frac{\theta}{2}\sin\frac{\psi - \phi}{2} \\ q_2 &= \sin\frac{\theta}{2}\cos\frac{\psi - \phi}{2} \\ q_3 &= \cos\frac{\theta}{2}\sin\frac{\psi + \phi}{2} \\ q_4 &= \cos\frac{\theta}{2}\cos\frac{\psi + \phi}{2} \end{aligned} \tag{3.27}$$

And θ,ψ,ϕ are Euler's angles representing successive rotations about the z, x' and z' axes. It turns out the time derivatives of the orientation parameters (q_1, q_2, q_3, q_4) can be expressed in terms of quaternions themselves and the angular velocities.

$$\begin{aligned} \dot{q}_1 &= \frac{1}{2}(-q_3\omega_x - q_4\omega_y + q_2\omega_z) \\ \dot{q}_2 &= \frac{1}{2}(q_4\omega_x - q_3\omega_y - q_1\omega_z) \\ \dot{q}_3 &= \frac{1}{2}(q_1\omega_x + q_2\omega_y + q_4\omega_z) \\ \dot{q}_4 &= \frac{1}{2}(-q_2\omega_x + q_1\omega_y - q_3\omega_z) \end{aligned} \tag{3.28}$$

$$\sum_{i=1}^{4} q_i^2 = 1 \tag{3.29}$$

Above equations can be solved explicitly for the quaternion values at the new time step in terms of the old values and the angular velocities at the midpoint of the timestep using time centered finite difference scheme.

3.4 Hybrid MPI+OpenMP Design Considerations

Message Passing Interface (MPI) is the industry-standard Application Programming Interface (API) for message passing across distributed-memory machines and often incorporates with the idea of domain decomposition. Open Multi-Processing (OpenMP) is a standard API for parallel programming on shared-memory architectures that parallelizes serial code by adding directives to instruct the compiler how to distribute workload at the data level. A hybrid HMP+OpenMP model is considered in this section to fully leverage the potential of mutliprocessing clusters. This combination of models makes use of data distribution and explicit message passing between cluster nodes, as well as shared memory and multithreading within nodes.

3.4.1 Data Abstraction and Motivation for the Use of Data Structure

Three levels of abstraction are used in the binning algorithm: blocks, bins, and grains. To begin, the computation domain is partitioned into blocks, each of which is "owned" by a computing processor. The block is then separated into equal-sized bins with a length equal to or greater than the diameter of the largest grain in the assembly. The number of processors is chosen to minimize the total area of communication between the computational sub-domains. The bin length can be expressed mathematically as $R + \Delta R$, where R is the largest diameter of the grain in assembly and ΔR, is determined empirically to strike a balance between the size of bins and the maximum number of grains that can reside in a bin, as increasing the bin size has the same effect as decreasing the total number of bins required to partition the domain. As a primitive task unit, each bin contains grains and communicates with 26 neighbors to detect and resolve contact and force. Finally, grains are assigned to a given bin based on the coordinates of their mass center in relation to the bin and block sizes.

In the realm of clustered system, it is prudent to define a fixed number of grains in a bin, as conveying all data at once is far more efficient than sending a fraction several times due mainly to data arrival latency. As a result, the bin size should be fixed to minimize border/ghost layer communication overheads. The simplest method is to calculate the size of the bin in advance and estimate the maximum number of grains that can fit within.

This technique grows linearly for large numbers of grains of roughly the same size and simple shape, but is less efficient for assembling arbitrary shaped avatars with more dispersed gradation. As a result, a binning algorithm that explicitly models all bins and allocates each grain to its proper bin is memory intensive and prone to memory management problems. Furthermore, from a practical standpoint, the number of bins may exceed the number of grains if the majority of bins are empty, resulting in resource waste. A more elegant solution is to use a linked-list abstraction to map grain-bin associations, so that grains are sequentially indexed by a unique number and only the exact number of total grains is maintained. This technique efficiently implements the belonging relationship between bins and grains, the neighbor list for each grain, and the ghost bins using four lists in $O(1)$ operations. The MPI implementation section discusses this algorithm in greater detail.

3.4.2 Border/Ghost Layers Communications

In a binning algorithm, each grain is uniquely labelled using a hash on their coordinates and based on the bin-size. A carefully chosen cutoff distance guarantees that every grain could only interact with grains in 26 neighbor bins, as even the largest grain cannot extent across more than two bins. However, a boundary grain may extend across into neighbor sub-domains, and this requires each sub-domain to maintain a copy of remote grains to correctly account for the interactions of boundary grains. This creates a halo region which is an extended layer of bins outside the boundary bins and records all updates from neighboring sub-domains. The process of exchanging border information is termed border/halo communication. The purpose of border/halo communication is to update boundary information for sub-domains because each processor only understands its own space and the associated grains. Therefore, before proceeding to a new timestep, boundary grains in the border/halo area must be updated with the latest translational velocity, angular velocity, rotation in the global frame, and center mass location. The new code packs information beforehand and minimizes communication overheads effectively. The amount of data required to update remote data from halo regions is constant and small; hence it can be packed to send collectively. Note that grains' shear history in border/halo layers does not need to be transmitted because those grains are computed on their host processors.

3.4.3 Across-Block Migration

Across-block migration refers to a circumstance when a grain enters or leaves its host block. If a grain migrates across the block border, one sub-domain needs to delete it while another needs to add this grain. In contrast to the border/ghost communication where data can be packed and communicated together, the across-block migration shall be considered individually for different grain because the size of the message being transferred is different. A naïve algorithm analogous to border/ghost migration is to construct a spatially outward extension from each processor's border (Yan & Regueiro, 2018b). The size of the extended layers is independent of the size of virtual bins, and it is determined by the velocities of the

grains and the time step used in the current time increment. Each processor should check its extended layers to see if any of its grains move into the computational domain. If yes, the corresponding processor should send such grains to the interested processor and delete them from its own space. However, this is not the best solution for across-block migration because a grain can essentially move to any other processor. Besides, the grain velocity is difficult to estimate for some dynamic problems.

In general, a three-dimensional DEM simulation falls into two main categories: static or quasi-static problems and dynamic problems. The velocity range of the first category could be reasonably estimated, for the second category, it is more difficult to determine the position that a grain could reach. The idea of extension layer also faces technical difficulties in its implementation. In the binning algorithm, blocks that mapped on sub-domains might have different spatial locations and hence create many edge cases. Moreover, if a history-dependent tangential contact model is chosen, the shear history ought to be sent as well. It is less efficient to send and receive complex information of varying length from specific senders to specific receivers using point-to-point communication APIs (primitive send and receive functions) in MPI implementation. Moreover, this is very likely to cause communication deadlocks. Although such an issue can be addressed using BoostMPI, which runs on top of MPI implementation, or using one side communication features of MPI3. The proposed code adapts highly tuned collective functions, MPI_Alltoall() and. MPI_Alltoallv() which makes the code more readable because all processors call the collective functions; consequently, the same code is applicable for all processors.

3.4.4 Dynamic Workload Distribution

The workload distribution is almost uniform among processors in simulating semi-static load problems such as the triaxial compression test even though grains may consist of various numbers of nodes on their reconstructed surface because the grain assembly is arranged in a relatively uniform and compact manners. Therefore, although the specific number of grains residing in a bin varies, the total amount of working load in a sub-task is similar, and the approximate cost can be evaluated in advance. Typically, static domain decomposition works well for homogeneous condensed matter, but it is not as good for inhomogeneous grain distributions due to performance-limiting work imbalances. For highly dynamic modeling of debris flow or earthquake simulations, characterized by large grain displacement between integral steps, grains can move and enter arbitrary sub-domains frequently, causing the number of grains residing inside a fixed-size bin to fluctuate as well.

The proposed parallel scheme aims to be applicable for wide ranges of simulations and adopts an adaptive re-binning scheme for highly dynamic problems. This scheme decomposes the latest computational domain and redistributes resources to best balance the current geometrical configuration. The subroutine can be either called at each timestep or after specific numbers of timesteps depending on the problems. The re-binning step has the same time complexity as the standard step. It does not affect the performance because dynamic adjusting and re-assigning grains at the beginning of a timestep essentially calls

the same subroutine as across-block migration, it also skips border/halo communication. The fundamental idea behind adaptive re-binning is to treat the boundary sub-domains as infinitely large temporarily. Any grain about to leave the computational domain will still behold in the boundary sub-domains. After a certain amount of timesteps, the whole computation grids will be regenerated, referred to the maximum and minimum coordinates of the entire bunch of grains in each direction. An alternative of adaptive dynamic binning is the quad-tree algorithm (a recursive coordinate bisection algorithm), which divides the domain into flexible bin size, counts the number of grains within a bin, and emphasizes the load balance of each bin. While the dynamic decomposition approach can improve load balancing of MPI instrumented simulations, it is limited by its focus on balancing the number of grains rather than the actual workload among processors (Yan & Regueiro, 2019). This can shift computational effort to individual processors, which leads to additional undesirable work imbalance.

3.4.5 Modeling Flexible Membrane in DEM

Both fixed and moveable rigid boundaries are readily modeled for parallel implementation because the geometries of such boundaries are simple and can be modeled as a single entity. However, a deformable boundary is required to initiate and develop strain localization, irregular boundaries like a flexible membrane in a triaxial simulation are challenging to parallelize. In a conventional triaxial test, a cylindrical specimen is enclosed by a flexible latex rubber membrane that can stretch or contract in response to local deformation. To replicate the latex rubber in DEM, packed spheres were arranged in a hexagonal pattern and deform with relation to their surrounding spheres, as illustrated in Figure 3.4. Contact bonds were allocated to the membrane spheres to keep them connected while allowing relative motions between spheres to be unrestricted. The contact bond strength between membrane spheres was set to a value that was sufficiently weak to allow for complete development of the shear band and volumetric change while remaining sufficiently strong to enclose the specimen. Unlike the temporary springs used in the soft-contact model of inter-grain interaction, membrane spheres are permanently connected via normal and tangential springs. The normal spring secures the spheres in place as the membrane stretches or contracts, while the tangential spring maintains the hexagonal pattern of the spheres. The stiffness of these springs is much smaller than the grain-membrane contact stiffness, allowing the membrane to deform freely mimicking the latex rubber.

Numerically, an external force is applied to each sphere in accordance with the confining pressure specified. In essence, the magnitude of the applied force on the membrane spheres is equal to the product of the confining pressure and the triangular region's area. Moreover, the applied force is perpendicular to the region and pointed inward to the specimen. For a triangular mesh element T, we can calculate the outward normal $\mathbf{n}$ and the directional area $\mathbf{S}$. Thus, according to the divergence theorem of Gauss, the volume of the specimen can be calculated:

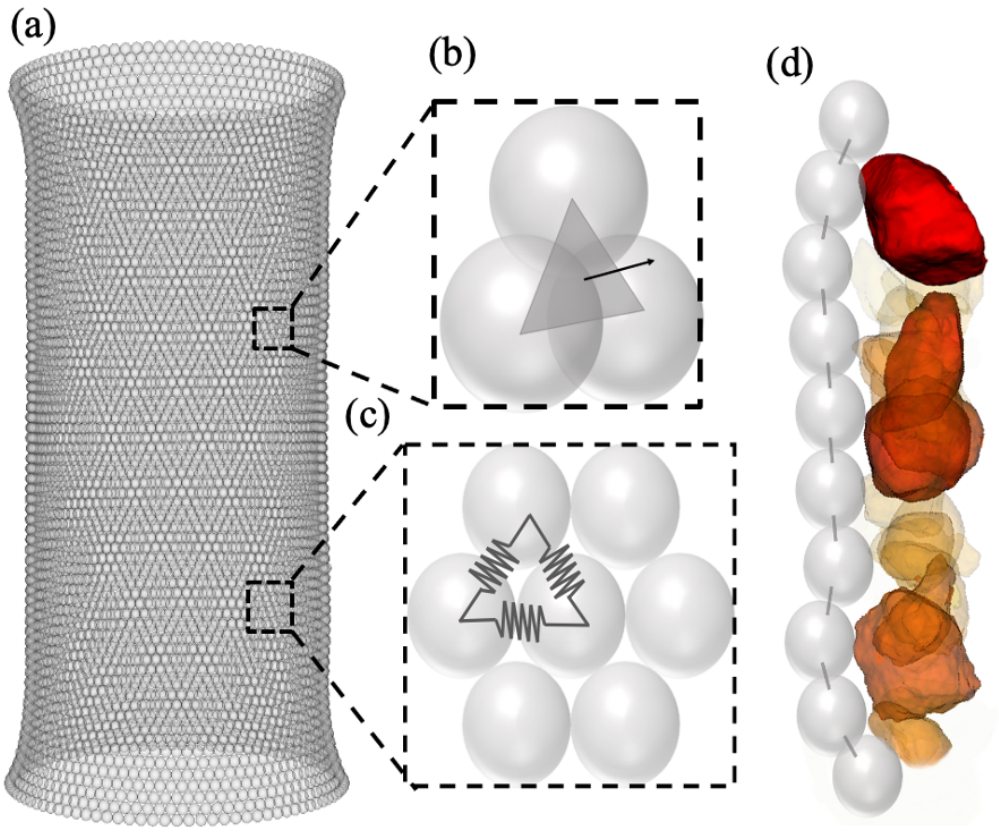

Figure 3.4: (a) Illustration of flexible membrane, $\sim 15,000$ balls. (b) Confining pressure normal to faces. (c) Hexagonal patterns of bonded spheres. (d) Membrane-grain interaction.

$$V = \iiint_{\Omega} dV = \oint_{\partial\Omega} \mathbf{S} \cdot \mathbf{n} d\Gamma \approx \sum_{T \in \partial\Omega} \mathbf{S}_T \cdot \mathbf{n}_T \tag{3.30}$$

Due to the simple geometry of the membrane elements, interactions between grains and the membrane are straightforward and as follows: an avatar node determines whether the distance between the node and the sphere center is less than the radius of the sphere and determines the amount of penetration as well as the normal direction. However, this procedure has two limitations. To begin, the grain of interest should iterate on all membrane spheres and perform a bounded circle check to identify nearby spheres, before refining the force resolution phase on discretized nodes. While sphere-grain interaction is considered to be notably less computationally intensive than inter-grain interaction, this process is still $O(n^3)$ and requires a significant amount of computing power. Second, the flexible membrane is designed to conform to the geometric configuration of the specimen and strike a balance between confining pressure and resistance forces, which is proportional to penetration depth

and is dependent on contact stiffness. However, once the membrane sphere enters the grain completely, all distances between discretized nodes and the sphere center exceed the radius of the sphere, and the sphere is no longer considered to be interacting with grains. Consequently, the spheres will continue to move inward and rapidly distort the entire membrane. This issue can be resolved by introducing a set of sophisticated criteria that classify the position of the membrane sphere relative position of the grains. However, this is far from an efficient or elegant measure. As a solution, we implemented grain-membrane interaction similarly to inter-grain interaction by temporarily moving spheres in the principal frame of the grain of interest and reading penetration from its LS grid. This modification increases efficiency in two ways. Only one lookup of the LS table is required to complete a pair of grain-membrane interactions. This process can be applied in a variety of different scenarios without modifying the code or establishing arbitrary rules because decoding LS values from a fixed underlying grid is reliable and accurate.

Directly partitioning and parallelizing a flexible membrane according to the hash of coordinate is challenging for the following reasons: 1) assignment of a membrane sphere to one sub-domain hashed on its spatial coordinate is complicated as boundary deforms locally, and it is tedious to track and update spheres that are about to migrate across sub-domains. In addition, the hexagonal connectivity of a sphere should also be maintained before and after the migration; 2) it is possible that a membrane sphere and its contacting soil grains belong to different sub-domains. As a result, to correctly capture this relationship, the implementation must memorize the coordinate of a membrane sphere and its spatial relationship with contacting grains; and 3) although the information exchange between membrane spheres and contacting grains involves border communication and cross-domain migration and has a great similarity to the information exchange between grains, this kind of message passing occurs much more frequently and retards the computation.

3.4.6 Memory Management

Our initial implementation of parallel code duplicated one copy of all grains for each processor, which turns out to be an acceptable design when the problem size is modest. The critical factor in this design is to optimize simulation speed, as each processor can handle all updates to grain velocities and rotations without reloading the morphology data, and it is simple to implement. Because reading data from main memory is approximately hundred times slower than reading data from cache in modern systems, preserving data locality is critical for performance. However, this notion is ultimately infeasible for a variety of reasons. To simulate large problems, the increased number of processors similarly increases the size of the overall data file, producing four distinct issues: 1) the combined effect puts a strain on the memory, leaving little space for calculation; 2) in a clustered computing system, multiple processors share a finite amount of memory, and we may be unable to allocate sufficient memory to each processor, resulting in underutilization of the computing resources available; and 3) during the initialization phase, the code must read all data files many times, which turns out to be a significant expense, and 4) this approach defies the goal of

parallelism because another common reason to parallelize serial code, aside from speeding up execution, is to partition the data between processors when it is too large to fit on a single processor. Clearly, retaining a copy of data regardless of its size violates this design concept. With these constraints in mind, the solution is to read morphological files only once and assign grains to one of the processors based on their coordinates. When a grain is about to migrate to another computational sub-domain, the new host processor consults the database and regenerates the avatars using the most recent velocity, rotation, and friction information. Rather than releasing the memory associated with a grain that migrates to another sub-domain, we mark it as unused and retain it in memory in case it migrated back later, reducing the number of times we looked up the database. This procedure was not invoked frequently during the simulation of quasi-static type problems, resulting in a negligible amount of additional overhead.

3.4.7 Limitations of the Proposed Parallel Implementation

Our implementation regards the whole computational domain as a big box and cuts it into identical cuboid sub-domains. Therefore, the underlying connectivity pattern for message passing is determined. As a result, the implementation details such as linked list-like data structure, halo communication, and grain migration are heavily reliant on this specific yet sophisticated sub-domains configuration. Fortunately, our implementation is tuned to utilize an arbitrary number of processors that are available and to suggest the best domain granularity. However, a major drawback is that it cannot properly handle problems where the distribution of grains across the domain is inhomogeneous, in another word, domain sparsity. For instance, while dealing with a cone-shaped container, we must still create the computational domain as a box large enough to encompass the cone. As a result, some sub-domains are left empty, resulting in resource waste.

The second drawback of this implementation is that it cannot use as many processors as possible because the cutoff size of a bin must be larger than the maximum equivalent diameter of a grain assembly. In addition, the cutoff size is chosen so that the average number of grains assigned to one processor is larger than a prescribed value. Although this implementation can guarantee that a processor is saturated by at least a certain amount of workload, they are also reasons why some of the cores being used are halted at times. Furthermore, the domain decomposition strategy induces work imbalance among processors due to different amounts of additional interactions with ghost bins in border layer. The center sub-domain is totally encircled by others and is influenced by ghost bins from all six faces, whereas a corner sub-domain has just three ghost bins. Consider a cubic sub-domain consists of n^3 bins, and the number of bins involved in force resolution is $(n+2)^3$ for a center sub-domain, but only $n^3 + 3n^2 + 3n + 1 = (n+1)^3$ for a corner sub-domain, with $n = \text{Domain Size}/(\text{Bin Size} \times \text{Processors})$. The work ratio $(n+2)^3/(n+1)^3$ decreases as the number of processors increases and increases as the number of bins per sub-domain increases. Finally, the implementation is not universally applicable and ought to be modified for different contact models, boundary conditions, and sorting algorithms, although the code

can handle many types of problems. As previously stated, parallel implementation treats various boundary conditions quite differently, and the implementation primarily relies on the current linked-list-like data structure to index and sort grain and to map the relationship between grains, bins, and blocks. Hence, both the halo layer communication and the grain movement are intrinsically linked to the specific sorting algorithm.

3.4.8 Programming Environment

The principles of Object-Oriented Programming (OOP) were followed in the design and programming of the code. Specifically, various classes are designed to model the practical concepts and objects that exist in a DEM simulation system: grains, bins, and exchange information packages. The Standard Template Library (STL) is heavily used, such as vector and set, to ensure code robustness and performance. The Eigen library is used to facilitate matrix-matrix or matrix-vector multiplication and transfer mathematical operations naturally. In developing MPI functions of three-dimensional LS-DEM, both border/ghost communication and across-block migration are accomplished using highly tuned API functions.

3.5 Parallel Implementation Details

3.5.1 Parallel Implementation of the Membrane

We explored two methods for parallelizing membrane type boundaries. Here we address their benefits and drawbacks. Parallelizing a flexible membrane is more complex than parallelizing a rigid boundary that does not deform locally. The two approaches to membrane simulation in a parallel environment are based on the concept of addressing local grain-membrane interaction in distinct computational subdomains specific to the corresponding host processors. We note that, the concept of directly decomposing flexible membranes into different processors was quickly abandoned due to the inefficiency of tracking grain-membrane interactions, updating spheres, and maintaining hexagonal connectivity across processors. For example, a single large grain may interact concurrently with multiple membrane spheres, but each sphere may belong to a different processor; additionally, the hexagonal pattern should be preserved whenever a sphere migrates to an adjacent processor.

Our first parallel implementation (Figure 3.5) utilized an additional processor to exclusively handle grain-membrane interactions. Before the contact detection and force resolution phases, all normal processors communicate most recent positions of the grains to the membrane processor, so that the grains in the membrane processor are up to date. In membrane processors, grain-membrane interactions are resolved concurrently with inter-grain interactions, and grain forces and moments are communicated back to normal processors. The most advantageous aspect of this strategy is that the time spent on computing grain-membrane interaction is separated from the inter-grain interaction, thereby reducing the serial portion

of code. However, the implementation is slightly more complicated and doubles memory consumption due to the additional copies of grains required by membrane processors. Alternatively, our second strategy (Figure 3.6) does not employ an additional membrane processor but rather operates grain-membrane interaction immediately following force resolution; each processor is equipped with its own membrane, and only those spheres interacting with grains are activated. At each time step, processors broadcast and gather the forces of activated spheres from different sub-domains prior to advancing a time step. This strategy appears to be more elegant and simpler to implement; it also results inn a more qualified parallel code because all processors execute the same task. However, interaction between grains and membranes becomes a necessary component of all processors. We prioritized speed when choosing between the two strategies and discovered that the performance of the two versions is dependent on the number of grains being modeled; when the problem size is small, the membrane processor can complete grain-membrane interactions faster than normal processors and vice versa if the problem size becomes larger.

We benchmarked our parallelized code with up to $700,000$ grains and investigated both strong and weak scalability for the domain decomposition strategy designed specifically for our application. Our code can simulate problems of comparable size and consumes less than 5% of the computing resources required by the embarrassingly parallel code. The various performance-critical subtleties are optimized so that communication overheads are kept to well below 1% of total computing wall time in favorable configurations, that is, each processor is saturated with sufficient, balanced work. Due to the low communication overhead, the weak scalability is nearly perfect, i.e., same computing time for twice larger problems using twice more resources. By contrast, the strong scalability rapidly degrades as more processors are used. This is because the sub-domain located in the center of entire computational space requires more ghost bins from neighboring sub-domains to complete halo layer exchange, force resolution, and grain migration than its edge and corner counterparts. In other words, if the bin size is significant in comparison to the total domain size, the domain decomposition strategy is inherently load imbalanced. If we instead compute and consider the true workload of each sub-domain, the measured performance gain matches the expected value reasonably well. Again, this issue is unique to our application of simulating a densely packed grain system in a quasi-static setting, where the bin size has to be larger than the largest grain diameter in an assembly and not insignificant in comparison to the overall specimen configuration.

3.5.2 Minimizing Memory Consumption and Preventing Memory Leaks

The C++ class to present a grain used in LS-DEM consists of a large LS table which implicitly represents a grain, hundreds of nodes seeding on the reconstructed surface along with shear histories on each. Most of these data remain unchanged and only small amounts of data have to be communicated among sub-domains. As a result, it is wiser not to store a

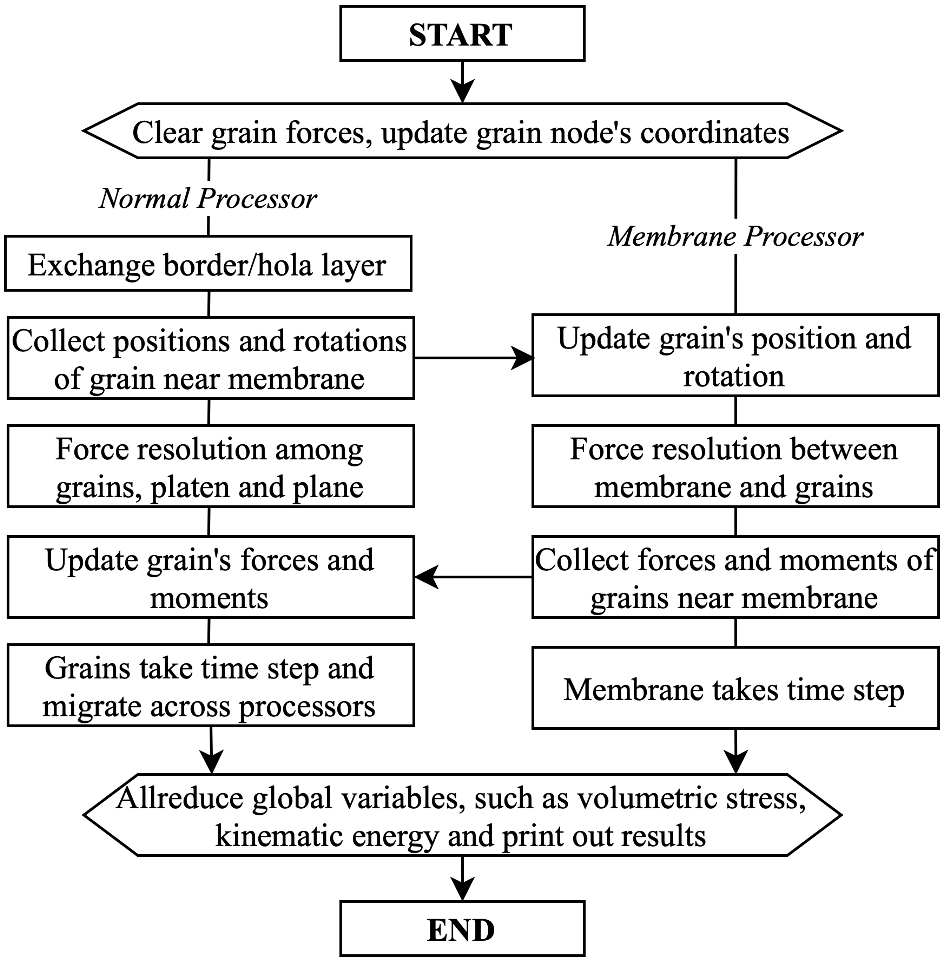

Figure 3.5: Flowchart of parallel strategy using extra membrane processor to specifically handle membrane-grain interactions.

whole grain object in the bins, and only index them instead. The new code minimizes the amount of memory consumption in different ways.

One of the crucial facts is that the number of grains can be smaller than the number of bins, which is somewhat counter-intuitive since bins are conceptually more extensive abstraction than grains. For a mini grain assembly in our test, 74 grains were distributed across a $100 \times 100 \times 100$ computational domain. The largest diameter of the assembly is 20, hence the cutoff distance is chosen as 20; this creates 125 bins partitions. Therefore, instead of building 125 bins and assigning grains hashed on their mass centers, it is more efficient to maintain an array of 74 grains and keep track of their bin indices. Although contact detection, resolution algorithm, and time step integral working directly on the grain array, the bin structure is still required to assist message packing in MPI communication.

The rigid body assumption significantly reduces the number of messages to exchange. Un-

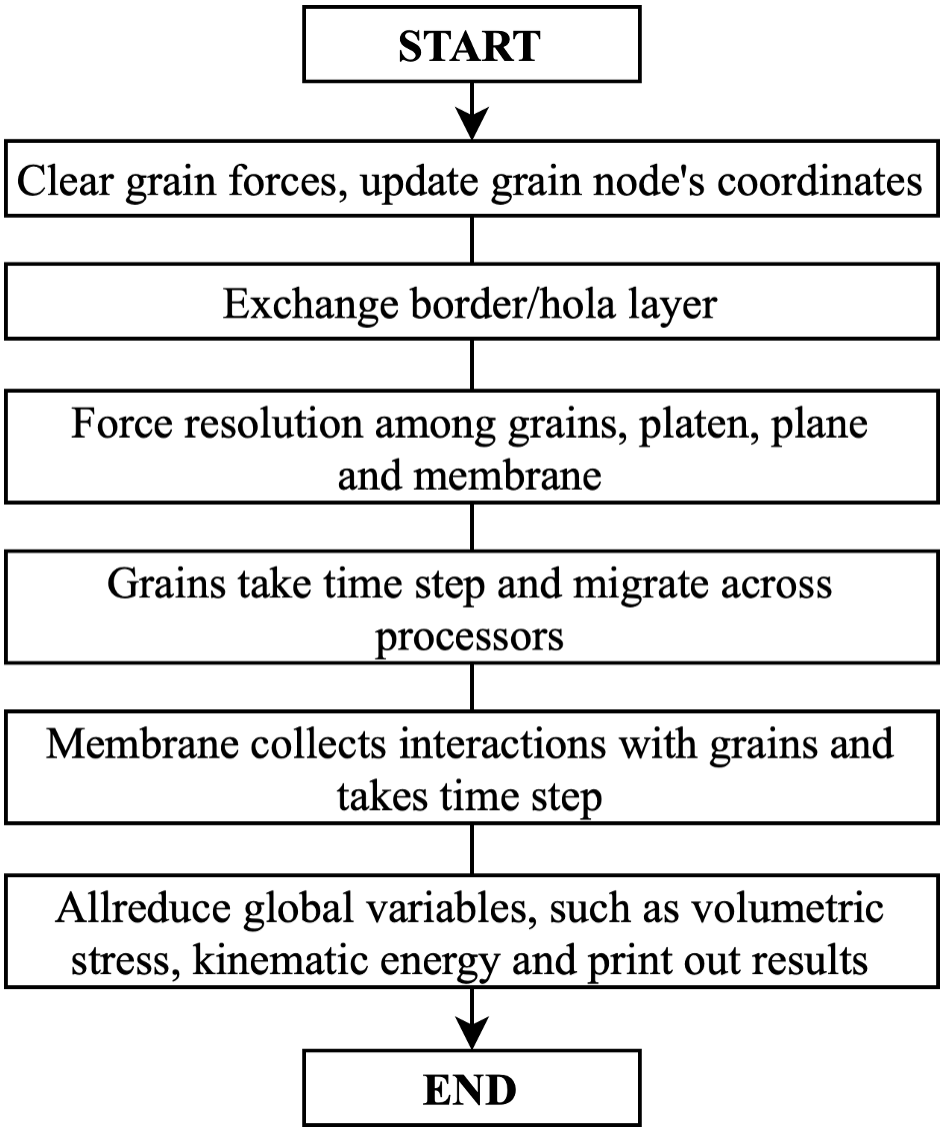

Figure 3.6: Flowchart of parallel strategy that membrane-grain interactions are handled in the same way by all processors.

der this assumption, the information required to update grain status for the next timestep only consists of the positions of mass center, rotations with respect to the global frame, plus the translational and angular velocities. Those are quantities needed to update boundary grains in the outmost bins of a sub-domain via border/halo communication. Since both LS table and discretized nodes' positions relative to the center are unchanged during simulation, we have another implementation option to update grains: regenerating the grain from its morphological file with updated quantities. That is: after border/ghost communication, a new grain can be generated from corresponding morphological file with the freshly computed location, quaternion, and velocities. This alternative requires less memory for a very large problem but needs to access disk memory more frequently. Compared to the border communication, across-block migration is much more complicated attributed to the usage of the history-dependent tangential contact model, where the shear history should be brought together with the migrating grains because moving to other processors does not imply grains have separated from their contacting neighbors.

To avoid memory leakage, containers in Standard Template Library (STL) such as vector and set are heavily relied on. The smart pointer feature is used as much as possible whenever it applies. While raw pointers cannot be completely avoided due to MPI's incompatibility with smart pointers, they are produced and discarded with special care. Every processor keeps a copy of the full grain list to map bins and grains, which is a memory-efficient strategy. Besides, the copy constructor and assignment operator are overwritten to avoid memory issues. Eigen and STL also inherently implement deep copy and provide memory protection.

3.5.3 Code Simplification and Edge Case Avoidance

One of the essential rules of MPI implementation is that as many processors as feasible accomplish the same task. To simplify implementation and avoid edge cases, each block/sub-domain is wrapped in a layer of padding bins (the number of bins along each dimension is increased by two, so that all bins within the sub-domain are regarded spatially similar, that is, each has a full set of 26 nearby bins. At the domain decomposition level, our code treats each processor identically by adding additional computational domains in such a way that each processor is associated with sub-domains of equal size. Furthermore, the number of processors used is determined to ensure that the communication area between MPI ranks is kept to a minimum. Then the code may dynamically select the optimal domain granularity based on both domain topology and resource availability. Moreover, the code establishes a limit value to ensure that each processor works on a minimum number of grains to conceal communication overhead; this value is established based on the prototype test results and is dependent on the average number of nodes used to discretize avatars. The code also takes advantage of MPI built-in functions like MPI_Dims_create(), MPI_Cart_create() and MPI_Cart_shift() to arrange processors into a grid, which allows MPI to automatically handle edge cases when communicating with neighbors, for example, when each processor

is instructed to send data to their left neighbor whereas there is no left neighbor for the left-most processor.

3.5.4 Reducing Computational Effort

To further minimize computation time, the proposed code makes use of Newton's third law and successfully reduces computation time by a factor of two. In the ideal scenario, where the grain surface is continuous and integrable at any location, the interaction force between master and slave grains is the opposite of the interaction force between slave and master grains while maintaining the same magnitude. However, due to the discrete representation employed in LS-DEM, the magnitudes of forces and reaction forces are not same. As a result, the code always applies the force exerted by the grain with the lower index to the grain with the higher value. Even though this measure can only guarantee that the resulting forces are identical within a sub-domain and is inadequate at the boundary, where forces are always exerted from the halo layer to the inner grains, it reduces randomness due to thread-level parallelism and domain decomposition, which keeps the differences between tests within an acceptable range. Newton's third law requires that only half of the adjoining bins be iterated. As illustrated in Figure 3.2, the code takes into account 13 neighboring bins, including all nine bins above the target bins and four bins at the same level as the target bins. Any 13 bins can be chosen, and the algorithm chooses these 13 bins since the additional interactions between grains and boundary grains are reasonably straightforward to handle. The code avoids handling edge cases as a result of padding. In a shared-memory system, the grain-grain interaction is entirely symmetrical because the computational domain and message exchange are not partitioned. This is not true for parallelism in distributed memory machines, because only 'real' grains are taken into account and updated when a real grain interacts with grains in the halo region. For instance, a bottom grain in the upper sub-domain should interact with ghost grains in the lower sub-domain, but only forces exerted by ghost grains on non-ghost grains are taken into account.

3.5.5 Position Reasoning and Linked-List Data Structure

A naive approach to binning iterates over bins to detect contacts and resolve forces. A more elegant and efficient technique used in our algorithm uses linked lists to map relationships between grains and bins and runs in time complexity $O(1)$. This approach enables a simple encoding and decoding of a bin ID associated with a certain grain with $O(1)$ complexity, and vice versa. The following example introduces four connected lists and is illustratetd in Figure 3.7.

(1) The first array:

```
int* bins = new int[num_of_bins];
```

has a length of the number of bins and is initialized as all -1. The value that is stored in each element of bins is the first grain number in that bin. For instance, bins[23]=13 means the first grain in the 23-rd bins is #13, bins[4]=-1 means there is currently no grain stored in the 4-th bins.

(2) The second array

```
int* binList = new int[num_of_grains];
```

has a length of the number of grains and is initialized as all -1. The value stored in each element of binList is the bin ID that the grain belongs to. For instance, binList[13]=8 means grain #13 is stored in the 8-th bins.

(3) The third array

```
int* grainList = new int[num_of_grains];
```

has a length of the number of grains and is initialized as all -1. The value stored in each entry of grainList is the next grain ID stored in the same bin as the current grain. For instance, grainList[13]=17 means the next grain which was stored in the same bins as grain #13 has index #17; similarly, grainList[17]=-1 means there are no more grains stored after grain #17. As a more comprehensive example, instructions bins[23]=13; grainList[13]=17; grainList[17]=9; and grainList[9]=-1 describe a search path if one is interested which grains are stored in bins #23, the first grain is #13 which followed by #17 and #9. There is no sequential order among grains, the one is added merely to assist neighbor searching and interaction computation.

(4) The fourth array

```
int* belongToRank = new int[num_of_particles];
```

has the length of the number of grains and is initialized as all 0. The value that is stored in each element of belongToRank is in the enumerator $\{0, 1, 2\}$, 0 means the grain is not associated with the current rank, 1 means the grain is residing inside the associated sub-domain, 2 means the grain is a ghost grain related to current rank. For instance, belongToRank[2]=0 implies the 2-nd grain does not belong to the current rank.

3.6 Comparison Between Different Implementations

The original LS-DEM code developed by Kawamoto et al. (2016), it implements a hybrid MPI+OpenMP in a naive way and achieves good speed-up using 480 cores. However, the performance gains were obtained by naively arranging all computing resources in for-loops to increase throughput (the number of iterations accomplished) at a time; consequently, the computational complexity of the code is still $O(n^2)$. Here we introduce two MPI parallelisms

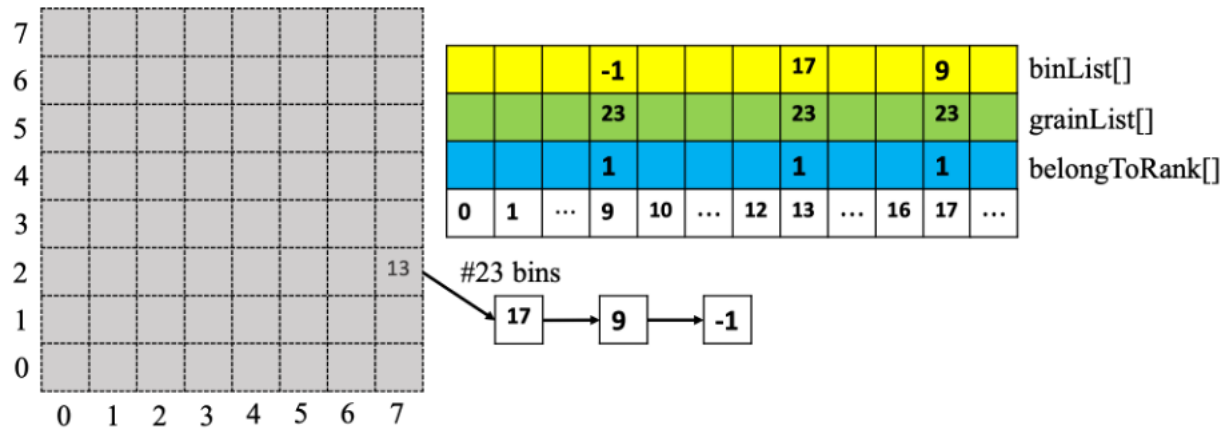

Figure 3.7: Illustration of grain search using linked list data structure

that successfully reduce the computational complexity to $O(n)$ with the binning algorithm. The difference between these two versions is that one implementation iterates bins while the other iterates grains for contact detection, force resolution, and grain update.

The numerical experiment was run with the GNU C compiler on Lenovo NeXtScale nx360m5 nodes of the Savio system at UC Berkeley, which has 2 Intel Xeon Haswell processors with 12 cores per processor @2.3GHz. Each core has two 512-bit-wide vector processing units, four hardware threads. Each core has 256KB L1 cache, and every four cores share an 8MB L2 cache. The goal of this experiment was to measure both the strong and weak scaling of MPI implementations and to observe the performance improvement by integrating $O(n)$ algorithm and using parallel techniques.

3.6.1 Original Implementation of LS-DEM

The primary code developed by Kawamoto et al. (2016) uses simple hybrid MPI+OpenMP implementation, which does not employ binning algorithm or tree algorithm to reduce the computation complexity. It loops over all the grains and simply parallels the code using OpenMP directives, the load balance is improved by specifying a dynamic loop schedule. Each processor learns the update grains through collective function MPI_Allreduce(), gathering information from all computing units and distributing it back. Although collective communication and identifier MPI_IN_PLACE could reduce some memory motion, and border/halo communication phase could also be removed because all processors have a copy of grains, there is still a large waste of computational efforts because this algorithm requires $O(n^2)$ computational complexity, and all grains are participating in the global all-to-all communications. Not surprisingly, the existing parallelism outperforms the proposed implementation with the binning algorithm for a small number of testing grains (74 grains) because

it only communicates once at each timestep. However, this algorithm is memory-demanding and begins losing efficiency as the problem size grows. Moreover, the original code does not protect false sharing or data race. This implies two grains may simultaneously interact with the same grain and compete to write on the third grain's memory. Besides, grains are not spatially sorted to enhance cache locality.

Nevertheless, the original LS-DEM code is fast enough for most of applications, it can model a triaxial compression test on approximate $60,000$ grains within one day. This means the code design is successful although naïve. The main limitation is that the code requires each processor to own one copy of all of the data and hence occupies substantial amount of memory if MPI is merely used in a brute force way to accelerate the code. Even though LS-DEM's contact algorithm has constant time complexity with respect to grid resolution, a large memory footprint nonetheless may lead to increased computation time due to cache misses and limit the number of grains that can be simulated. For example, a $40 \times 40 \times 40$ reconstructed avatar requires 64,000 LS values to be stored.

3.6.2 Two Binning Algorithm Implementations

The flowcharts of the two parallel implementation introduced here are depicted in Figure 3.8 and Figure 3.9. They show four major flow components for both naïve binning algorithm implementation and the one implemented in this work. Both implementations share four major components, but differ from each other in terms of memory efficiency, amount of interaction computation, border/halo communication, and across-block migrations. These four steps are not equally weighted or even proceed by different times during the simulation. The computational complexity is effectively reduced from $O(n^2)$ to $O(n)$ for both implementations. Bins from each sub-domain are isolated from other processors and updated with information available in the host sub-domain. Therefore, they do not participate in the message passing phases. The false sharing or data race issue is also avoided in both implementations because every grain is only associated with and managed by one processor at a time.

3.6.3 Border/Ghost Communication Algorithm

At the beginning of a timestep, the border/halo layers communicate with neighbors. Usually, a sub-domain needs to exchange information with its neighbors through six surfaces, twelve edges, and eight vertexes. In the new code, this step is simplified and reduced to three sequential steps. Taking advantage of the blocked and synchronized MPI_Sendrecv() function, the message passing procedure is guaranteed to happen in a right order. MPI_Sendrecv() is well-suited to send and receive a message simultaneously because it is specifically designed to circumvent deadlocks. The border/ghost communications algorithm is depicted with a two-dimensional illustration (Figure 3.10 and Figure 3.11), and the 3D algorithm is analogous. Two steps are sufficient for a two-dimensional border/ghost communication.

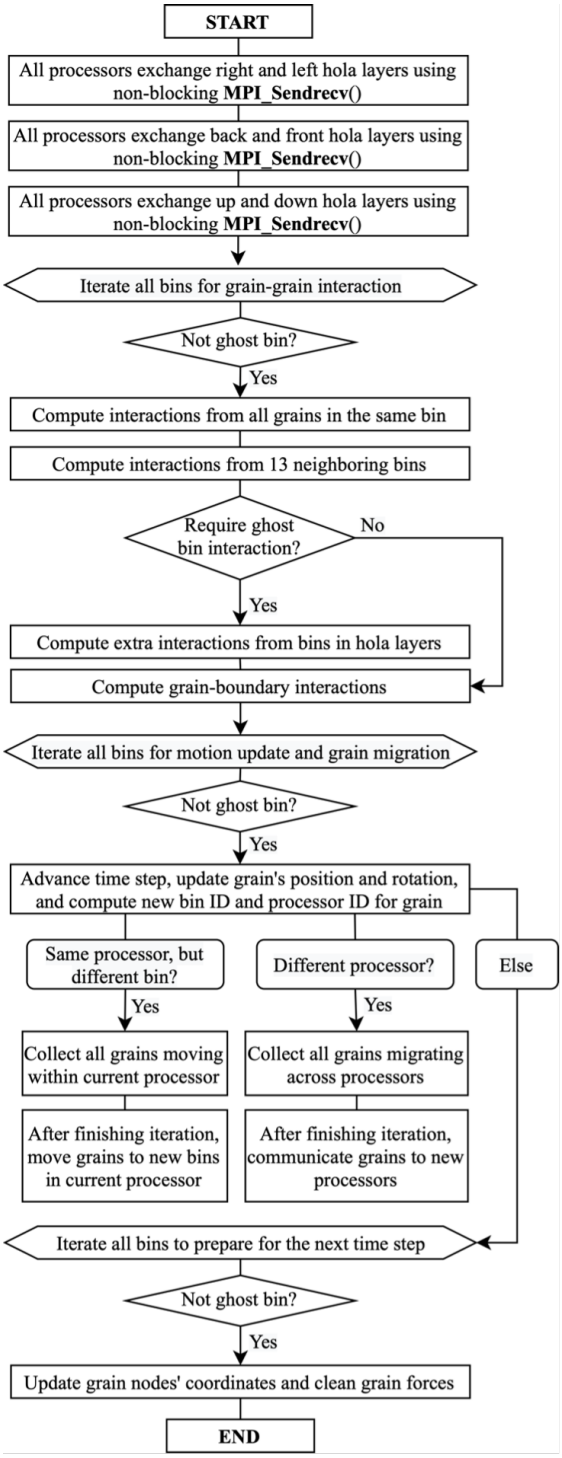

Figure 3.8: Flowchart of naïve binning algorithm where bins are basic elements in iteration.

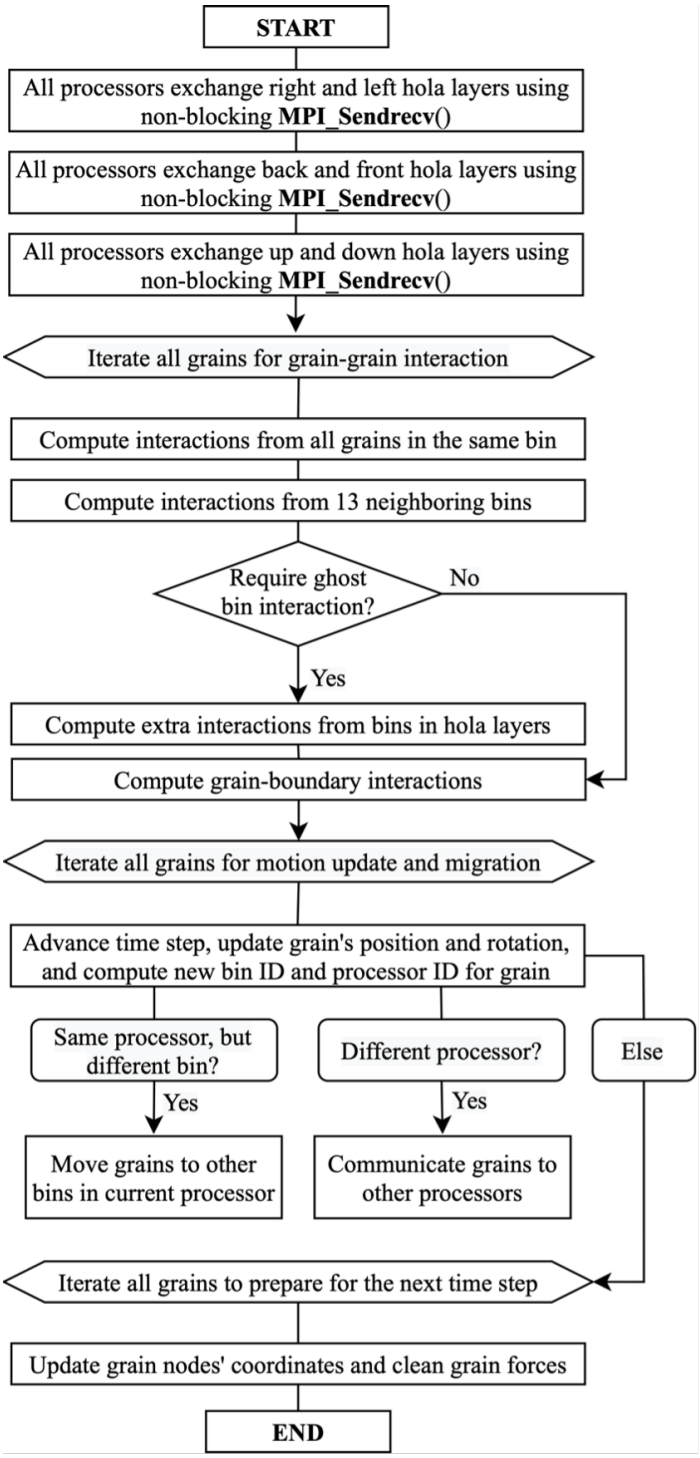

Figure 3.9: Flowchart of proposed binning algorithm where grains are basic elements in iteration.

Consider a two-dimensional computational domain divided into six sub-domains. Each sub-domain is further divided into a 3×3 bins matrix (grey areas in the plot). Every sub-domain also maintains a copy of boundary bins of remote processors so that a layer of ghost bins is used to wrap the original 3×3 bins matrix and form a 5×5 matrix. The proposed algorithm demonstrates that two sequential calls of MPI_Sendrecv() are enough to update all ghost bins, and only three sequential calls are required for a three-dimensional simulation.

As the first border/halo communication step (Figure 3.10), each processor sends its rightmost non-boundary bins (marked as type-1 bins) to update the left boundary bins of its right neighbor. Simultaneously, the current processor accepts a message from its left neighbor and updates the left boundary bins. MPI_Sendrecv() synchronizes this processor and completes this procedure at the same time. Similarly, each processor sends its leftmost non-boundary bins (marked as type-2 bins) to update its left neighbor's right boundary bins. The current processor then accepts a message from its right neighbor and updates the right boundary bins. The edge cases where the left or right neighbors do not exist are automatically handled because the proposed code has organized processors and constructed a Cartesian grid using MPI_Dims_create(), MPI_Cart_create(), and MPI_Cart_shift().

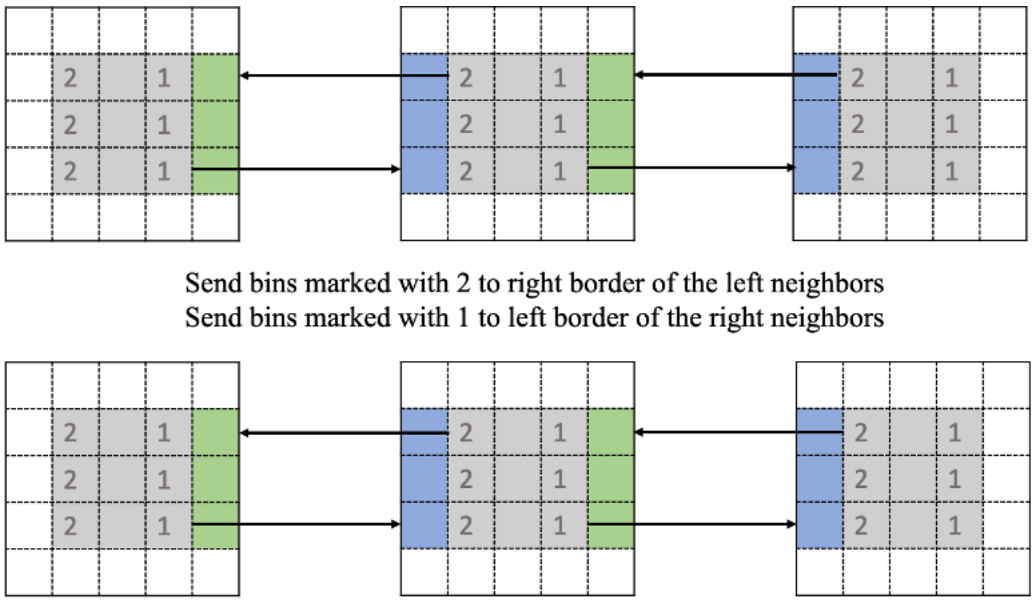

Figure 3.10: Border/ghost communication step one, exchange left and right layers.

In the second border/ghost communication step (Figure 3.11), each processor sends its upmost non-boundary bins (marked as type-3 bins) to update top neighbor's bottom boundary bins. Simultaneously, the current processor accepts a message from its bottom neighbor and updates the bottom boundary bins. After these two sequential steps, the corner bins in the ghost area are all updated. One could consider the procedure to update corner bins a two-step motion: first, send corner bins to the left and right neighbors, and the left and right neighbors, after receiving the corner bins, re-direct the corner bins to top and bottom.

More complicated three-dimensional border/ghost communications are implemented using precisely the same logic and three steps. The left and right ghost bins are updated first, then the back and front ghost bins, finally the top and bottom ghost bins.

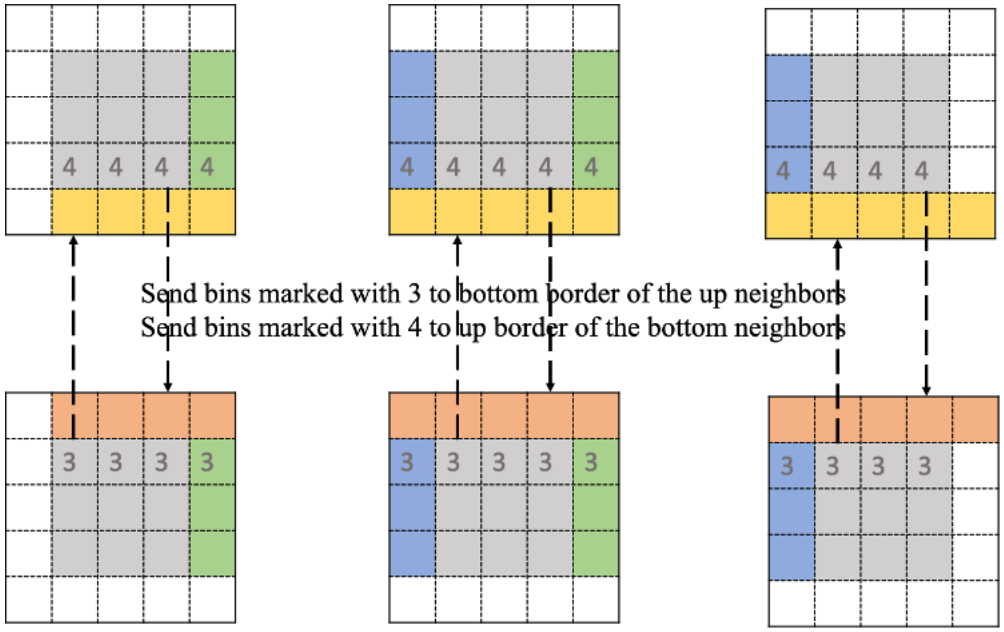

Figure 3.11: Border/ghost communication step two, exchange up and bottom layers.

3.6.4 Iterate Over Bins vs. Iterate Over Grains

The following stage involves computing the interactions between grain pairs. The straightforward implementation iterates over bins and computes inter-grain interactions for grains that reside there, starting with interactions between grains in the same bin and progressing to interactions between grains in nearby bins. This is a simple yet inefficient technique, as bins may be empty. In contrast, we introduce a novel force resolution strategy which iterates over grains and does optimum amount of work. The inter-grain relationship and grain-bin relationship are mapped using linked-lists with time complexity $O(1)$. Carrying an envelope calculation, a strategy that iterates grains instead of bins could reduce the amount of analysis by an order of magnitude. The resulting code scales linearly and computations are directly proportional to the number of grains. In contrast, whenever a grain is queried, the naïve implementation code requires to iterate over all bins to find the target. Therefore, the naïve implementation is computationally more expensive and brings about redundancy. Also, consider a situation in which one grain is about to migrate, it has not already left the host sub-domain but obtain a new bin ID that differs from the current one. But this operation cannot be immediately performed until all bins are checked because it is possible

that the grain could move to unchecked bins and be recalculated. In the end, both implementations using data structure abstraction should consider extra interactions from ghost grains since a sub-domain does not understand ambient information at the clustered system.

3.6.5 Implementation of Across-Block Migration

The motion of a grain is updated at the third step, and across-block migration may occur at the fourth step. Both naive binning algorithm implementation and our new implementation using data structure abstraction adopt a same algorithm. This is a relatively expensive step because the contact history is also brought along with the migrated grain if the history-dependent tangential contact model were applied. Although each migrated grain is considered individually, necessary information such as translational velocity, angular velocity, mass center location, and grain rotation can still be packed collectively to exploit overlapping and reduce communication latency. To scatter and gather migrated grains to other processors, collective operations like MPI_Alltoall() and MPI_Alltoallv() are heavily relied upon. It is also possible to only implement fundamental MPI_Send(), and MPI_Receive() functions with neighboring 26 processors if the timestep is small enough so that grains will not move across an entire processor. However, this approach is not generalizable, and using fundamental MPI_Send() and MPI_Receive() in nested loops would easily cause deadlocks. The across-block migration approach is illustrated below. We pack and communicate the common quantities first, and send grain specific information (shear forces, normal shear direction, and contact grain ID) individually. This procedure is illustrated in Figure 3.12 and Figure 3.13 and expanded with more detailed in following.

Packer is a container to store necessary grain information, the underlying data structure of packer is:

```
vector<vector<basic_grain_property>> packer(num_of_proc);
```

The basic grain properties have six components: grain ID, number of nodes on the grain, position, quaternion, translational velocity, and angular velocity. When a grain leaves the current sub-domain and enters another, the associated necessary grain information is pushed back into the related packer[new_proc_id]. The contact history is contained in three containers, which is defined similarly as :

```
vector<vector<Eigen::Vector3d>> packerNodeShears(num_of_proc);
```

tracks the vector of tangential forces on each node, the length of the element is changing as the number of nodes seeded on a grain is not constant. The second quantity of a contact history is the grain ID that a discretized node is contacting with. This is stored in

```
vector<vector<int>> packerNodeContact(num_of_proc);
```

The third quantity is the unit normal direction of a node in the principal body frame of the last time step. This is stored in

```
vector<vector<Eigen::Vector3d>> packerNodeNormals(num_of_proc);
```

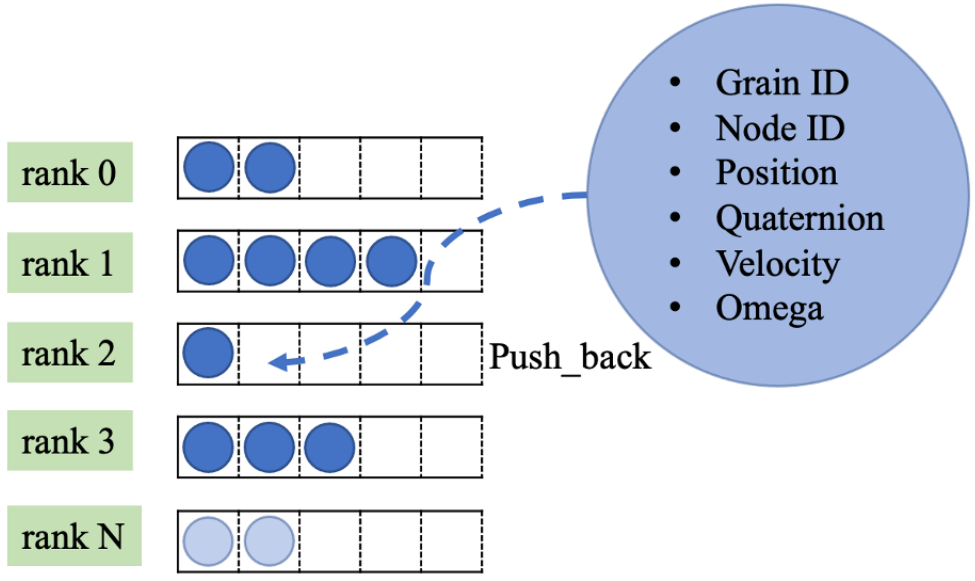

Figure 3.12: Illustration of data containers for basic grain information in across-block migration.

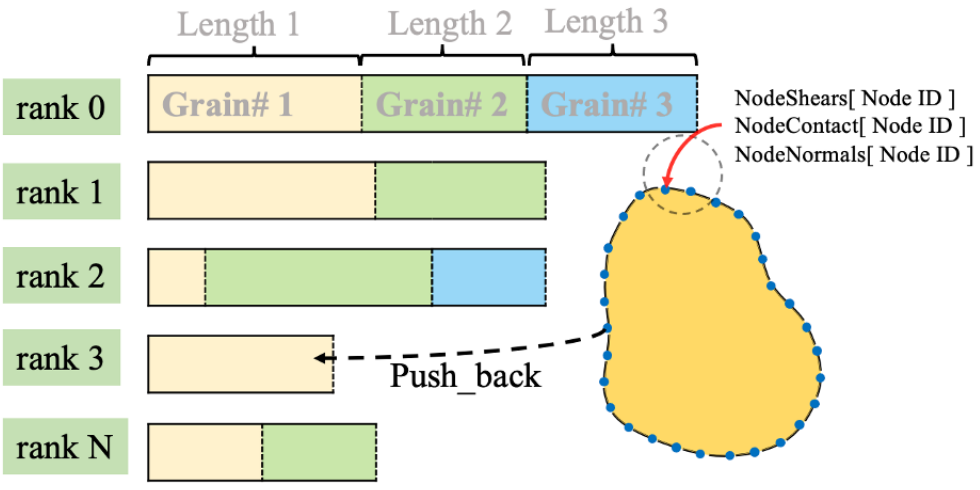

Figure 3.13: Illustration of data containers for contact history in across-block migration.

The data communication is achieved using the MPI collective functions MPI_Alltoall() and MPI_Alltoallv(). MPI_Alltoall() is a collective operation for the case where each processor sends distinct data to all other processors. The j-th block sent from processor i is received and placed in the i-th block of processor j's buffers. MPI_Alltoallv() adds flexibility to MPI_Alltoall() in that the location of data to be sent is specified by send_buff_displacement

and placed in the location specified by recv_buff_displacement. The effect of all-to-all operation, MPI_Alltoall(), is equivalent to a matrix transpose operation as illustrated below.

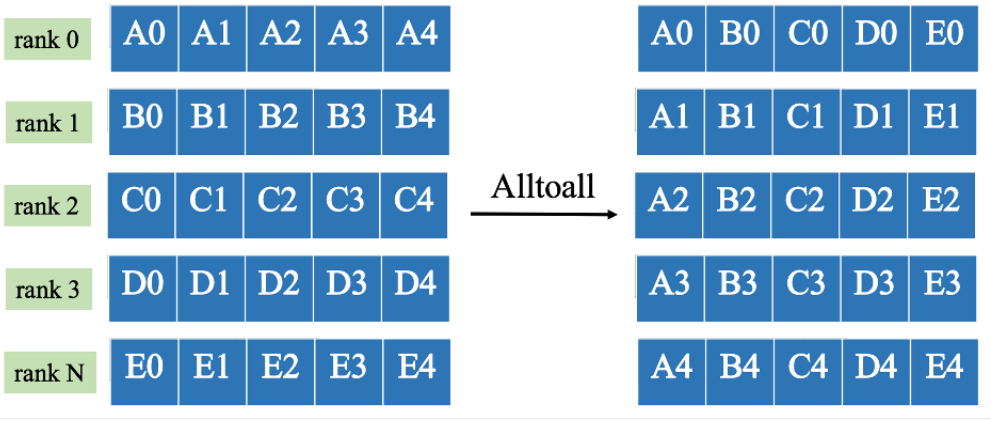

Figure 3.14: Illustration of the mechanism of MPI_Alltoall().

3.7 Numerical Experiments

3.7.1 Numerical Experiments on a Small Dataset

A series of small numerical experiments was performed on the Savio HPC at UC Berkeley EECS. The objective was to observe the performance improvement by integrating an $O(n)$ algorithm and using parallel techniques, as well as to study communication overheads suffered by small-sized problems. The problem sizes are multipliers of 74 from duplicating existing avatars with positions shifted to avoid overlapping. The computational domain in each numerical test is cubic, for instance, 592 grains are constructed by 8 copies of existing 74 grains and shaped into a $200 \times 200 \times 200$ domain. The problem setting is simple: domain boundaries are modeled as undeformed planes; a grain is not allowed to leave the domain and would be bounced back if it intends to do so. For simplicity and for work balance, grains are subjected to a random force at each timestep. The bookkeeping and adaptive binning are switched off when the performance is benchmarked.

Figure 3.15 shows the performance speed-ups by varying the number of processors. All speed-ups were measured relative to a serial run, which is equivalent to an MPI simulation with a $1 \times 1 \times 1$ decomposition. Each processor works on its own smaller region and maps to the closest physical memory to maximize memory bandwidth usage and avoid bandwidth limitations. Consequently, a well-balanced MPI implementation has high efficiency and keeping the communication cost low. However, there is a point beyond which more MPI processors lead to an increased MPI traffic due to the communication overheads and stalls or deteriorates scalability. In addition, increasing domain granularity also increases

the area of inter-processor interfaces through which boarder layers are exchanged, and more ghost grains are cloned to create extra workload. An evenly distributed workload and suitable domain granularity help to hide communication overhead, i.e., if the major tasks were finished later than the communication, then a good linear speed-up is expected as the cost of communication latency is amortized. In general, increasing the number of grains would bring the scalability closer to the idealized speed up. This is because the CPU is saturated and the portion of simulation time spent on contact detection, force resolution and grain update are more significant compared with those used for communication. This also explains why the overall performance of larger testing sample is better than the smaller counterparts. However, with increasing number of processors, MPI traffic due to data migration and increased synchronization times would add up and dominate. One may consider increasing the number of processors is analogous to reducing the problem size. In this sense, a single parameter defined as the number of grains associated with one processor is more appropriate to optimize parallelism.

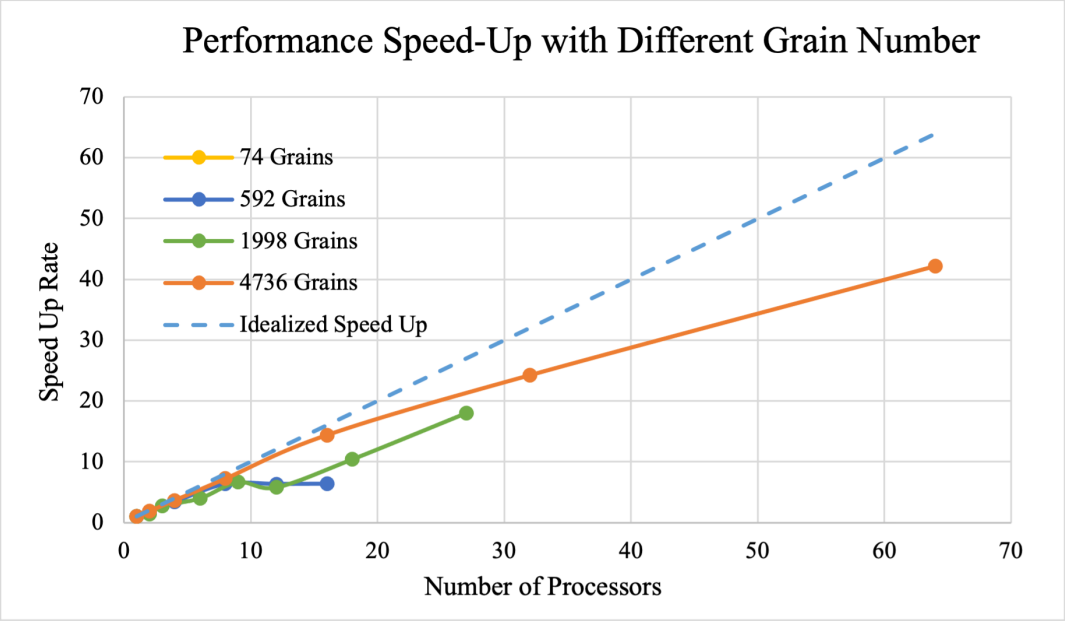

Figure 3.15: Comparison between obtained speed-up and idealized speed-up for a series of small numerical experiments, to study strong scalability.

Figure 3.16 shows that the running time decreases as the number of processors increases. Our code displays considerable speed-ups by evenly distributing work to more processors. Reading from the plot, it takes 852 seconds to finish $1,000$ timesteps with a single processor, while this number drops dramatically to 20 seconds if the parallel techniques are used. However, the running time is not a purely decreasing function of the number of processors, this is because domain granularity also considers bin size apart from the availability of computational resource. For example, to simulate 1998 grains residing in a $300 \times 300 \times 300$ domain

with 2 processors and consider bin size 100. Then we can at best divide the computational domain into $100 \times 300 \times 300$ and $200 \times 300 \times 300$, and as the result, instead of experiencing 2 times speed-ups, we could only save one third of computation time (256 seconds using 2 processors vs. 356 seconds using single processor). In our application, the bin size is chosen being at least the largest equivalent grain diameter of specimen, this implies that the bin size is the minimum scale to which the computational domain could be divided, and we cannot gain parallel performance indefinitely just by using more computational resource, for instance, we cannot parallelize 74 grains in $100 \times 100 \times 100$ domain with bin size 100, and we cannot do better than 8 processors to speed up 592 grains in $200 \times 200 \times 200$ domain (the running time using 8, 12 and 16 processors should be same).

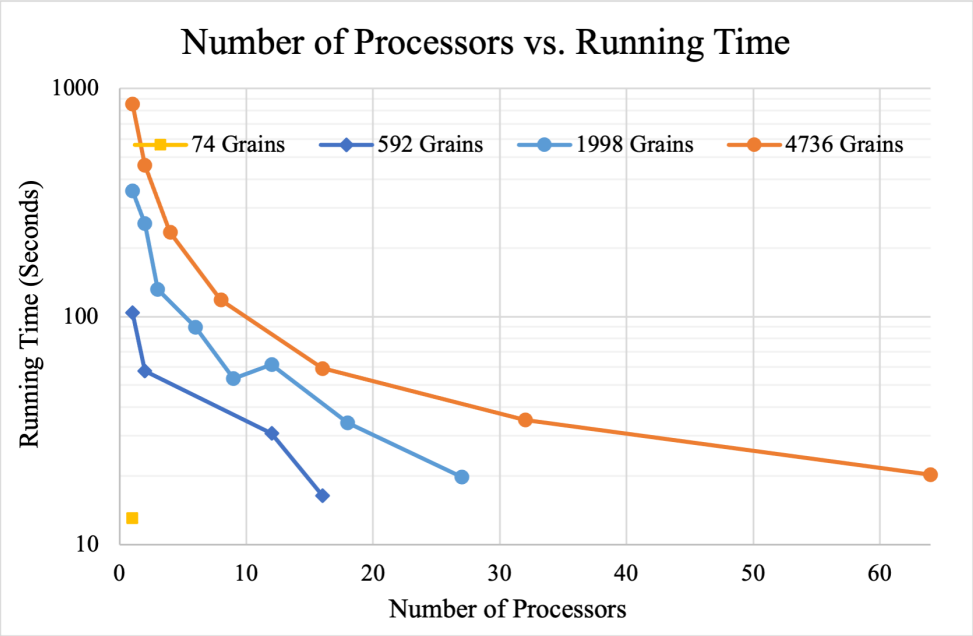

Figure 3.16: Relationship between running-time and number of processors for a series of small numerical experiments, to study of weak scalability.

Figure 3.15 and Figure 3.16 demonstrate the $O(n)$ computational complexity of domain decomposition. The general pattern is almost but not perfectly linear and there are several reasons account for this. Binning algorithm is based on the idea that inter-grain interactions would not occur if two grains are far enough apart, and DEM further assumes that only contact forces exist between grain pairs. As a result, the contact detection step in the proposed code acts merely as a filter to eliminate needless calculation and this process is nonlinear. For instance, the number of grains in a bin and its neighbors is not proportional to the number of grains that come into direct contact with the grain of interest. The computation cost depends on both grain geometry and spatial distribution of assembly; these properties vary substantially throughout the assembly. The force interaction phase is

distributed into tiered steps, only those pairs having passed the contact detection check are considered for force resolution, in which surface nodes are checked against the LS grid of the slave grain and the process would only be further refined if there exists a node indeed penetrates into slave grain's surface. Those steps minimize the amount of calculation, but also introduce sophisticated nonlinearity.

The running time breakdowns are displayed in Figure 3.17. All runtimes are decomposed into four categories, showing the amount of time spent in each phase. These statistics include: T_{force}: Time spent for contact detection and force resolution; T_{border}: Time spent in the explicit MPI border/halo exchange portion; T_{migrate}: Time spent in the grain moves to another bin or the grain migrates from host processor to new processor; and T_{update}: Time spent in parts of the code which manipulates the state of grains, meshes, and other computations such as numerical integration, and grain-wall interactions. Note that, the timings of each category is the maximum value over all MPI processors. Simulations of 4736 grains were chosen for analysis as this number was the amount to observe whether a processor is saturated or dominated by communication overheads. The inter-grain interaction accounts for more than 90% of running time implying that most of computation resources were used for inter-grain interactions. The border/halo exchange component is the function of the number of ghost grains, which is in turn controlled by the bin size and domain granularity. The grain migration is influenced by the magnitude of external forces, boundary conditions and grain densities. As can be seen, the running time for both MPI communication phases are trivial compared with force interaction computation this is because the external forces are too small to produce significant grain movements.

The time for synchronization has been implicitly included. Consider two portions of code where MPI communication takes place. The border/halo exchange phase is synchronized since all processors are synchronized at beginning of a new timestep. Synchronization for the across-block migration is accomplished by setting barriers. It seems that work imbalance appears during force resolution (and we will explain why later), which is supported by the fact that the across-block migration would take similar amount of time as force resolution after moving the MPI barrier between them. It is slightly confusing at the first glimpse because MPI_Alltoall() and MPI_Alltoallv() functions are highly optimized and are capable of handling peculiar situation, e.g, transmit zero-length message. The main reason appears to be that the MPI collective operations are blocked, i.e. faster processors must wait until all processors have completed previous phase to proceed the next. To relieve this issue, OpenMP directives are integrated in the new code, because work imbalance issue could be relaxed in the shared-memory implementation. Therefore, a hybrid type parallelism has potential to the best of both worlds: fully utilizing available computing resources and minimizing work imbalance via dynamic load scheme.

The running time breakdown may be deceptive because the time for force calculation is exaggerated due to work imbalance, such that the percentage of force calculation is raised. The widely adapted Amdahl's law (Rodgers, 1985) does not consider work imbalance and communication synchronization, a better metric serial fraction f proposed by Karp and Flatt (1990) is used to probe the effect of work imbalance. The speed up measured by running

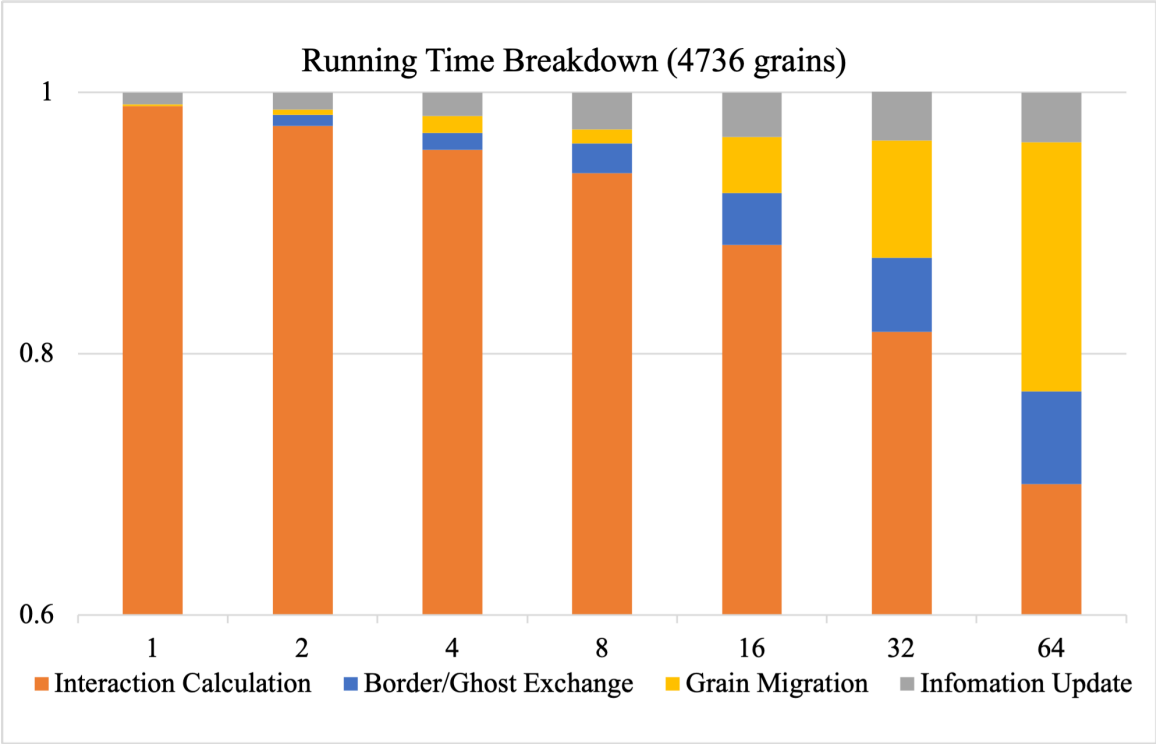

Figure 3.17: Running time decompositions for simulation with 4763 grains.

the same program on a varying number of processors is defined as:

$$s = \frac{T(1)}{T(p)} \tag{3.31}$$

where $T(1)$ is elapsed time with 1 processor. The issue of efficiency is related to price/performance, and is usually defined as:

$$e = \frac{T(1)}{pT(p)} = \frac{s}{p} \tag{3.32}$$

Consider the Amdahl's law which in the simplest form states:

$$T(p) = T_s + \frac{T_p}{p} \tag{3.33}$$

Where T_s is the time taken by the portion that must be run serially and T_p is the time in the parallelizable part. Then:

$$T(1) = T_s + T_p \tag{3.34}$$

If the fraction serial is defined:

$$f = \frac{T_s}{T(1)} \tag{3.35}$$

Then the Amdahl's law could be re-written as:

$$T(p) = T(1)f + \frac{T(1)(1-f)}{p} \tag{3.36}$$

Or in terms of speed up s:

$$\frac{1}{s} = f + \frac{1-f}{p} \tag{3.37}$$

The serial fraction could be solved as:

$$f = \frac{\frac{1}{s} - \frac{1}{p}}{1 - \frac{1}{p}} \tag{3.38}$$

The serial fraction f is useful because Amdahl's law is incomplete. Firstly, Amdahl's law assumes that all processors compute for the same amount of time, which implies that the work is perfectly load balanced. If some processors take longer than others, the speed-up declines and results in an equivalently larger serial fraction. Second, there is a missing term representing the overhead of synchronizing processors in Amdahl's law, which is a monotonically increasing function of number of processors. Since increasing overhead decreases the speed-up, increasing f is a warning indicator that the granularity is too fine.

Processors	Time (sec)	Speed-up (s)	Efficiency (e)	Serial fraction (f)
1	852.359	1.000	1.000	-
2	459.581	1.855	0.927	0.078
4	234.757	3.631	0.908	0.034
8	118.234	7.209	0.901	0.016
16	59.269	14.381	0.899	0.008
32	35.175	24.232	0.757	0.010
64	20.198	42.200	0.659	0.008

Table 3.1: Serial friction for simulations on $4,763$ grains.

The effect of work imbalance and synchronization is illustrated in Table 3.1. The results show that the computational effort was insufficient to saturate the CPU when more than 16 processors were used, thus large portion of running time was devoted to communication overhead. The efficiency is interpreted in a similar way as it experiences a rapid drop when more than 16 processors are used. The performance gain is most noticeable when all CPUs have enough computational demand and are arranged in a balanced domain granularity. In addition, the hardware configuration, particularly the cache and the memory bandwidth also play a role in determining the overhead.

3.7.2 Limitations (Strong Scalability) of Domain Decomposition strategy

More numerical tests were conducted to explore the code competence in simulating large number of grains. 42684 grains were constructed from relatively low-resolution images ($13-15\mu m$/pixel) resulting substantially more avatars and smaller morphological files for each. The total amount of computer memory used for this specimen is 2.5GB and the surface of each grain is discretized into 279 nodes on average. The entire computational domain is $600 \times 600 \times 600$, the largest possible grain radius is 40, the bin size is 50, and grains were subjected to random forces with mean magnitude about 10 times gravity to induce grain movement. The computational domain was partitioned in a way to minimize the communication area between adjacent processors. Even though the communication time tends to increase as the number of processors increases, it still takes less than 1% in all runs. Different sub-domains interact with different number of ghost bins depending on their location, e.g., a sub-domain in corners has less interactions than a sub-domain on sides, and both of them have less interactions than the one in the center. This causes work imbalance, and the ratio is associated with the number of processors and bin size. Intuitively, more processors and larger bin size yield more severe load imbalance. Table 3.2 lists the worst cases for each run. There are several layers of intricacies that determine the exact amount of work a processor undertakes. First, the total workload is not proportional to the number of bins because contact detection and force resolution are extremely convoluted and nonlinear processes, however, we can still assume this relationship is linear because the specimen is large and dense. Second, not all ghost bins are visited equally frequently because we avoid half computation redundancy by accepting that forces between a contacting pair matched in magnitude and opposed in direction. The accurate estimation of visited times for a ghost bin depends on both location of sub-domain and the symmetric pattern used for inter-grain interactions. Again, we skip the computation and assume half of ghost bins are visited, $\alpha = 1/2$. Moreover, it is possible that no sub-domain is completely wrapped by neighbors as this requires at least 27 processors to achieve that, in such case, determination of α is bit tricky. Finally, apart from interactions from inner bins to ghost bins, there are also reversed interactions from ghost bins to inner bins. This is because we only consider 13 out of 26 neighbors of an inner bin for force resolution, i.e., if we consider interactions between inner bins and their upper neighbors, we then need to incorporate interactions from bottom ghost bins for completeness. The maximum number of bins that a ghost bin could influence is 9 out of 13 (only consider half neighbors due to symmetry) if the ghost bin is below the bottom and close to the center, and the minimum number is 1 out of 13 if the ghost bin is at the corner. Therefore, another multiplier $\beta = 1/13 \sim 9/13$ should apply. The estimated workload is computed accordingly and the true code speed up is tabulated in Table 3.3.

As shown in Figure 3.18, the measured speed-up is close to the approximate mean speed-up. Super linearity is observed such that parallel efficiency is greater than one, which is not uncommon in benchmarking a parallel code because performance gain from augmenting total cache size overweighs communication overhead especially when the number of processors used

Processors	T_{force}	T_{border}	T_{migrate}	T_{update}	Theoretical #Bins	Actual #Bins	Difference #Bins
1	67083.43	0.00	0.00	82.30	1728	1728	0
2	32727.49	23.74	0.95	51.41	864	1008	144
4	17836.61	22.54	1.14	33.39	432	588	156
6	14422.88	21.09	0.94	41.72	288	504	216
8	12661.43	20.17	0.36	36.25	216	343	127
12	9261.44	16.47	1.13	37.68	144	294	150
16	6525.09	13.68	1.00	22.58	108	245	137
18	8814.76	15.02	1.21	27.10	96	252	156
24	7118.26	15.11	1.33	26.13	72	210	138
27	4896.77	18.87	1.54	20.67	64	216	152
32	4940.85	20.04	2.68	22.46	54	175	121
36	5545.15	13.70	2.35	19.05	48	180	132
48	4495.77	12.45	2.67	19.29	36	150	114
54	4185.16	16.18	1.17	17.90	32	144	112
64	3900.41	17.82	1.71	17.66	27	125	98
72	3431.75	11.82	3.34	27.36	24	120	96
96	3218.85	11.96	4.36	14.36	18	100	82
108	3486.49	11.32	2.36	13.04	16	96	80
144	3135.79	15.74	4.41	12.06	12	80	68
216	2758.50	13.19	6.29	10.02	8	64	56

Table 3.2: Runtime breakdown for simulations of $42,684$ grains in a $600 \times 600 \times 600$ domain, with bin size 50, simulations ran for $1,000$ steps, (time is in seconds).

is small. We should point out the domain decomposition is a natural and recognized parallel strategy for many problems beyond the scope of DEM modeling, our application encourages us to probe why the performance gain would cease at some point if we were further increase computing resources, even though the communication cost remained at low levels. This will no longer be an issue if the bin size is much smaller than the size of sub-domain and a better linearity is expected.

3.7.3 Weak Scalability of Domain Decomposition Strategy

High performance computing has two common notions of scalability: strong scalability and weak scalability. Strong scalability is defined as how the solution time varies with the number of processors for a fixed total problem size. Our series of tests implies that strong scalability does not hold for domain decomposition strategy if the bin size is significant compared to the simulation geometry. Despite the inherent load imbalance nature and extra workload presented by the domain decomposition, our code nevertheless creates very small amount of communication overheads. Therefore, it would be better to study the weak

Processors	Upper ($\beta = 9/13$)	Upper ($\beta = 1/13$)	Upper ($\beta = 5/13$)	N_{Lower}	N_{Upper}	N_{Mean}	N_{True}
1	1728	1728	1728	1.00	1.00	1.00	1.00
2	1036	947	991	1.67	1.82	1.74	2.05
4	618	522	570	2.80	3.31	3.03	3.76
6	546	413	479	3.17	4.19	3.61	4.65
8	367	289	328	4.70	5.97	5.26	5.30
12	323	231	277	5.35	7.50	6.25	7.24
16	271	187	229	6.37	9.24	7.54	10.28
18	282	186	234	6.13	9.29	7.38	7.61
24	237	152	194	7.31	11.40	8.90	9.43
27	245	152	198	7.05	11.39	8.71	13.70
32	198	124	161	8.72	13.96	10.73	13.58
36	205	124	165	8.41	13.92	10.49	12.10
48	172	102	137	10.05	16.98	12.63	14.92
54	166	97	131	10.44	17.89	13.18	16.03
64	144	84	114	12.01	20.69	15.20	17.20
72	138	79	109	12.48	21.77	15.86	19.55
96	116	65	91	14.93	26.46	19.09	20.84
108	111	62	87	15.51	27.80	19.91	19.24
144	93	51	72	18.57	33.73	23.95	21.40
216	75	40	58	23.11	42.87	30.03	24.32

Table 3.3: Estimated workload of different number of processors, and corresponding theoretical speed-up. N_{Lower}, N_{Upper}, N_{Middle}, N_{True}: lower bound, upper bound, mean, and measured speed-up.

scalability of our code to show how the solution time varies with the number of processors with a problem size per processor fixed. We designed and conducted a series of numerical simulations and tabulated in Table 4.1. The largest possible grain radius in the simulations is in the $30 \sim 40$ range, therefore, the bin size is 50 for all. All simulations ran for $1,000$ timesteps.

The sub-computation domain size was identical for all test series, and the number of processors was chosen to ensure the entire domain is decomposed into identical pieces. As illustrated in Figure 3.19, the serial code does not incur extra ghost bin interaction and therefore takes less time to finish, simulations run with 8 processors are slightly faster than others for similar reasons. Again, we are only able to count the number of extra ghost bins that are potentially involved but less sure about how intensely these bins are visited. For all other domain granularity using greater or equal to 27 processors, the computing times are close and displays a slightly increasing trend with processors due to communication overheads. This suggests that weak scalability still holds even though the strong scalability

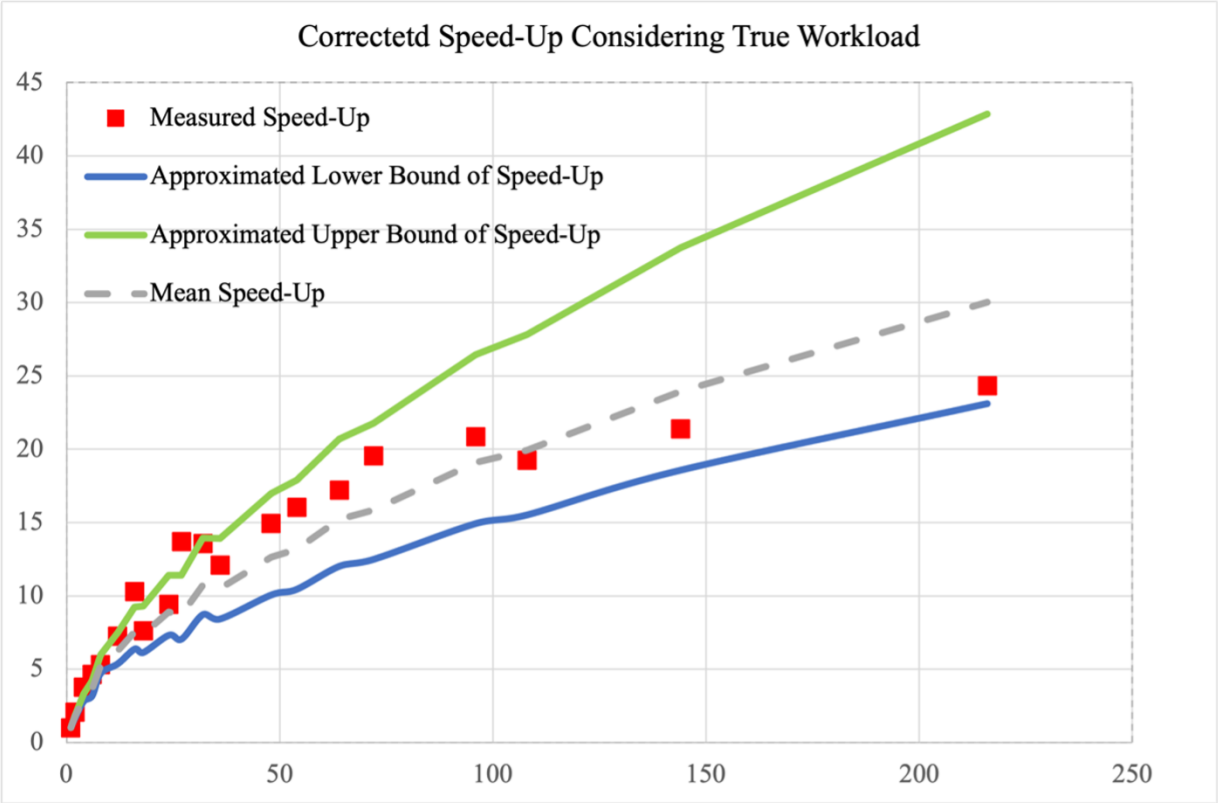

Figure 3.18: Corrected speed up relationship considering actual workload on processors, study of strong scalability.

does not.

3.8 Conclusions

A parallel code for 3D LS-DEM to model arbitrary-shaped granular materials has been designed and implemented in C++ building on existing LS-DEM framework developed by Kawamoto et al. (2016). It introduces the concepts of binning algorithm and effectively reduces the computational complexity from $O(n^2)$ to $O(n)$. The newly implemented code maps relationship between bins and grains with linked-list like data structure and considers MPI communication in two major parts: border/halo exchange and across-block migration. The time complexity of execution time, communication time and parallel overhead of proposed code are analyzed with regard to the amount of computational resources and the problem size. The result shows that the proposed parallel code has an excellent weak scalability numerically and has the potential for simulating large scale DEM problems with complex-shaped grains.

Grains	Domain size	Processors	Theoretical #Bins	Actual #Bins	Simulation Times(s)
Serie #1	sub-domain size: $200 \times 200 \times 200$				
12494	$400 \times 400 \times 400$	8	64	125	2501.74
99952	$800 \times 800 \times 800$	64	64	216.	2755.65
337338	$1200 \times 1200 \times 1200$	216	64	216	2984.45
799616	$1600 \times 1600 \times 1600$	512	64	216	2991.20
Serie #2	sub-domain size: $150 \times 150 \times 150$				
481	$150 \times 150 \times 150$	1	27	27	195.95
3848	$300 \times 300 \times 300$	8	27	64	321.87
12987	$450 \times 450 \times 450$	27	27	125	360.88
30784	$600 \times 600 \times 600$	64	27	125	386.64
60125	$750 \times 750 \times 750$	125	27	125	404.79
103896	$900 \times 900 \times 900$	216	27	125	399.62
164983	$1050 \times 1050 \times 1050$	343	27	125	398.47
246272	$1200 \times 1200 \times 1200$	512	27	125	410.13
Serie #3	sub-domain size: $200 \times 200 \times 200$				
1328	$200 \times 200 \times 200$	1	64	64	896.66
10624	$400 \times 400 \times 400$	8	64	125	955.22
35856	$600 \times 600 \times 600$	27	64	216	1081.27
84992	$800 \times 800 \times 800$	64	64	216	1185.02
166000	$1000 \times 1000 \times 1000$	125	64	216	1139.87
286848	$1200 \times 1200 \times 1200$	216	64	216	1133.78
455504	$1400 \times 1400 \times 1400$	343	64	216	1155.67
679936	$1600 \times 1600 \times 1600$	512	64	216	1172.96

Table 3.4: Domain angularity parameters for studies of weak scalability

The code also has high potential to only have negligible serial fraction and low parallel overhead for a larger, uniformly distributed assembly when executing on modern multiprocessing supercomputers. An important advantage of the MPI implementation is that the code is able to run on wide variety of parallel systems, including shared-memory computers and clustered systems. As a result, the code is highly portable. Future effort to simulate large-deformation problem, will require the implementation of the adaptive dynamic mesh or quad-tree algorithm to take advantage of the computational speed offered by code parallelization.

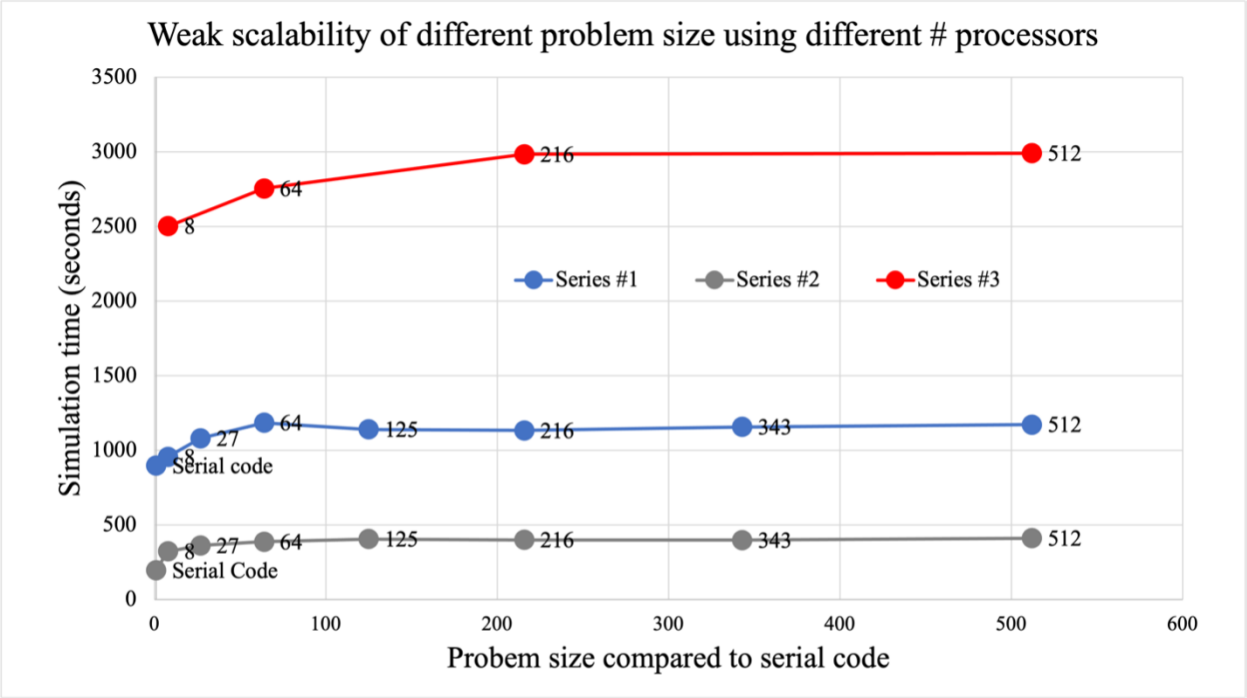

Figure 3.19: Simulation times for problem of different sizes and fixed workload for processors, study of weak scalability.

Chapter 4

LS-DEM Modeling of Naturally Deposited Sand in Triaxial Compression

4.1 Introduction

The motivation for the development of a more efficient LS-DEM code was our aim to numerically explore the mechanical behavior of dense assemblies of sand particles, such as occur in naturally deposited sands. In this chapter, we present a parametric evaluation of the conditions necessary for successful modeling of sand subjected to triaxial loading and we present a LS-DEM simulation of a full-scale triaxial tests with a systematically calibrated contact model. The grain morphology of the naturally deposited sand is reconstructed with great fidelity from XRCT images, and a realistic confining boundary is reproduced by modeling a flexible membrane composed of bonded spheres. The numerical triaxial test is then conducted by mimicking a conventional laboratory test.

4.2 Metric and Visualization

4.2.1 Friction Angle and Dilatancy Angle

Considering purely frictional material, the angle of friction is one of the principal material properties for granular assemblies. We can then resolve the stresses or, alternatively, compute the friction angle using the Mohr-Coulomb failure criterion.

$$\sigma_1 = \sigma_3(\frac{1+\sin\phi}{1-\sin\phi}) \tag{4.1}$$

Hence, the peak mobilized friction angle can be calculated as:

$$\phi_p = \sin^{-1}(\frac{\sigma_1 - \sigma_3}{\sigma_1 + \sigma_3})_p \tag{4.2}$$

Where the values of σ_1 and σ_3 are taken at the peak of the stress-strain curve. Similarly, the critical friction angle can be calculated as:

$$\phi_{cr} = \sin^{-1}(\frac{\sigma_1 - \sigma_3}{\sigma_1 + \sigma_3})_{cr} \tag{4.3}$$

Dilatancy is defined as the volume change associated with the application of shear stresses. An increase in volume, or expansion, is known as negative dilation, while a decrease in volume, or contraction, is known as positive dilation. The amount of dilatancy that a granular material can experience depends on the inter-grain interlocking, which depends on the fabric of the material. The dilatancy angle ψ can be estimated from the volumetric strain versus axial strain curve of a material subjected to triaxial compression with the following expression (Schanz & Vermeer, 1996; Salgado et al., 2000):

$$\psi = \sin^{-1}(\frac{\dot{\varepsilon}_v/2\dot{\varepsilon}_a}{2 - \dot{\varepsilon}_v/2\dot{\varepsilon}_a}) \tag{4.4}$$

In this work, we use the "mobilized" friction angle as one of the principal parameter in comparing the results of simulations.

4.2.2 Fabric Tensor

One of the important descriptors of granular texture is the orientational arrangement of the constituent grains which is generally termed fabric. Several descriptions of fabric have been suggested in the literature, each relying on the definition of a unit vector describing the direction of a certain microstructural property such as the grain major axis, branch vectors, contact normal. To shed light on the behavior of microstructure at locations of interest in the macroscopic domain, Satake (1982) and Oda et al. (1985) provided a contact normal-based tensorial formulation. They denoted $E(\mathbf{n})$ as the orientational density function of these microstructural quantities such that $\int_\Omega E(\mathbf{n})d\Omega = 1$ and $E(\mathbf{n}) = E(-\mathbf{n})$ due to the lack of intrinsic parity. A second order approximation of $E(\mathbf{n})$ gives rise to the usual second-order fabric tensor.

$$\mathbf{G} = \int_\Omega E(\mathbf{n})\mathbf{n} \otimes \mathbf{n} d\Omega \tag{4.5}$$

Or, in the discrete form:

$$\mathbf{G} = \frac{1}{N_c}\sum_{k=1}^{N_c} \mathbf{n}^k \otimes \mathbf{n}^k \tag{4.6}$$

Where N_c is the total number of contact points, and $\mathbf{n}$ describes the unit vector. Numerous investigations have demonstrated that the second order fabric tensor accurately approximates both the distribution of the orientation of the contact normals and the distribution of orientation of of normal and tangential contact forces. The primary direction of contact forces often coincides with the principal stress axis, whereas the principal direction of the contact normal fabric changes in response to shear to align with the principal stress (and strain rate) axis as well. This is also true for other classes of fabric tensors, such as those that describe the orientation of elongated particles. To further capture the highly anisotropic nature of packing during the course of shearing, the deviatoric second-order fabric tensor $\mathbf{F}$ characterized by the contact normal is defined as:

$$\mathbf{F} = \mathbf{G} - \frac{\text{trace}(\mathbf{G})}{3}\mathbf{I} \tag{4.7}$$

$$\mathbf{F}' = \frac{15}{2}\mathbf{F} \tag{4.8}$$

$\mathbf{F}$ is the deviatoric part of $\mathbf{G}$, $\mathbf{F}'$ is scaled so that the orientation distribution probability density function $E(\mathbf{n})$ of contact normal $\mathbf{n}$ in the global coordination system can be directly approximated by the second-order Fourier expansion as:

$$E(\mathbf{n}) = \frac{1}{4\pi}(1 + \mathbf{n} \cdot \mathbf{F}\mathbf{n}) \tag{4.9}$$

Another scalar anisotropy factor to quantify fabric anisotropy is defined as:

$$a = \frac{15}{2}\sqrt{\frac{3}{2}\mathbf{F} : \mathbf{F}} \tag{4.10}$$

4.2.3 Force Chain

Continuum mechanics is based on the assumption that applied forces are uniformly transmitted through a homogenized granular system. However, in reality, the interparticle force distributions are highly heterogeneous, and the applied load is transferred via a network of interparticle force chains. Within the granular system, this geometric disorder of particles results in inhomogeneous but structured force distributions. Buckling of these force chains results in deformation, and energy is dissipated by sliding at the clusters of particles between the force chains. Photoelasticity (Daniels et al., 2017; Abed Zadeh et al., 2019) and related techniques, as well as DEM, have been the most widely used approaches for detecting force chains. External loads are transmitted through a network of inter-grain contact forces. This is referred to as the strong force network, and it is the primary microscopic property that facilitates load transfer throughout the granular system. On the other hand, grains that are not part of the strong force network float like a fluid with very small loads at the inter-grain contacts, this phenomenon is known as a weak cluster. Tangential contact forces between grains in strong networks are significantly smaller than normal contact forces, with

magnitudes close to the frictional resistance between the grains. As a result, the frictional resistance between grains in weak clusters is almost completely mobilized, and the grains behave similarly to a viscous fluid. Therefore, deviatoric load is transmitted entirely through normal contact forces in strong networks, with only a minor contribution from weak clusters. The two networks exhibit fundamentally different features, with the strong network's forces decaying exponentially and the forces of a weak network decaying power law-like (Mondain-Monval et al., 1998; Antony, 2000). Stress propagates along characteristic directions, which correspond to the experimentally observed force chains in granular structures (Radjai et al., 1996). As deformation progresses, the number of grains in the strong force network decreases during shearing, resulting fewer grains to share the increased loads (Kuhn, 1999) and most of sliding grains are in weak clusters (Radjai et al., 1996). Additionally, when a strong force network buckles, it collapses, resulting in the formation of new force chains. As a result, the spatial distributions of the strong force network are neither static nor persistent. There is a nonproportional relationship between macroscopic friction angle of the grain assembly and interparticle friction angle. The percentage of sliding contacts decreases as interparticle friction increases (Thornton, 2000). As a result, interparticle friction works as a kinematic constraint for the strong force network rather than a direct source of macroscopic shear resistance. Specifically, strong force chains could not form if interparticle friction was zero, and the grain assembly would act like a fluid. Increased contact friction improves system stability and reduces the number of contacts needed to establish a stable condition. However, as long as the strong force network can be created, the magnitude of interparticle friction becomes irrelevant.

4.2.4 Macroscopic Stress and Strain

It is important to recognize that direct numerical simulations of the mechanical response of a macroscopic structure composed of a micro-heterogeneous material are virtually impossible, and that aggregate or macroscopic response within a statistically representative volume element is characterized using regularized or homogenized material models (Zohdi & Wriggers, 2001). The average macroscopic stress in a granular assembly is defined as:

$$\bar{\boldsymbol{\sigma}}^g = \frac{1}{\Omega_g} \sum_{c=1}^{N_c^g} \mathrm{sym}(\mathbf{f} \otimes \mathbf{x}^c) \tag{4.11}$$

Where $\mathbf{x}^c$ is the vector defined from the origin to the point of application of contact force $\mathbf{f}^c$ at the contact point c, with a total number of contact points N_c^g in the grain g. This is the discrete form of Christoffersen et al. (1981) and a more generalized average stress theorem considering body forces can be found in Zohdi and Wriggers (2004). In three-dimensional setting, two often referred quantities: the mean effective stress $\mathbf{p}$ and the deviatoric stress $\mathbf{q}$, are derived from this formula:

$$\mathbf{p} = \frac{1}{3}\bar{\boldsymbol{\sigma}}^g \tag{4.12}$$

$$\mathbf{q} = \sqrt{\frac{3}{2}\bar{\mathbf{s}}^g : \bar{\mathbf{s}}^g} \tag{4.13}$$

For the strain measurement in a conventional triaxial test, as modeled herein, the axial strain ε_a and volumetric strain ε_v can be determined based on the displacement of the loading platen and volume change of the encasement according to the following equations:

$$\varepsilon_a = -\int_{L_0}^{L} \frac{1}{l} dl = \ln \frac{L_0}{L} \tag{4.14}$$

$$\varepsilon_v = -\int_{V_0}^{V} \frac{1}{v} dv = \ln \frac{V_0}{V} \tag{4.15}$$

Where L_0 and V_0 are the initial height and volume of the isotropically consolidated specimen right before shearing, and L and V are the quantities of the deformed specimen during the course of shearing.

4.3 LS Modeling Configuration

4.3.1 Numerical Apparatus

A conventional triaxial test set up consisting of a cylindrical sample encased in a flexible membrane was modeled using LS representation. The end platens were attached to the flexible membrane and the top platen was allowed to move vertically and rotate about the ram-platen contact point. The use of a flexible membrane and a rotatable platen is critical for the development of shear bands, as without them, the deformation of specimen would be influenced locally by boundary geometry. Figure 4.1 is presented for ease of visualization. We found that the rotation of the upper platen is a significant factor in determining the type of shear band formed. The upper platen rotates as a result of the uneven force generated by the complex arrangement structure and interaction of encased LS avatars. Accordingly, the specimen deforms asymmetrically and, particularly at higher confining pressures, the samples tend to generate a single shear band. In comparison, if the upper platen does not rotate during shearing, the sample dilation is more uniform in the radial direction. The samples generate an X-shaped shear band in this case, see e.g. Liang and Zhao (2019) and Wu et al. (2021).

The triaxial test was performed in quasi-static conditions in two principal stages: isotropic compression and deviatoric loading. After the LS avatars were reconstructed in place and the material properties were assigned according to the contact-stiffness model, the grain assembly was subjected to isotropic compression with confining pressure of $\boldsymbol{\sigma_c} = 100$kPa over a large number of timesteps to reach mechanical equilibrium. At the end of consolidation, the middle portion of specimen slightly shrinks, and this is consistent with the experimental

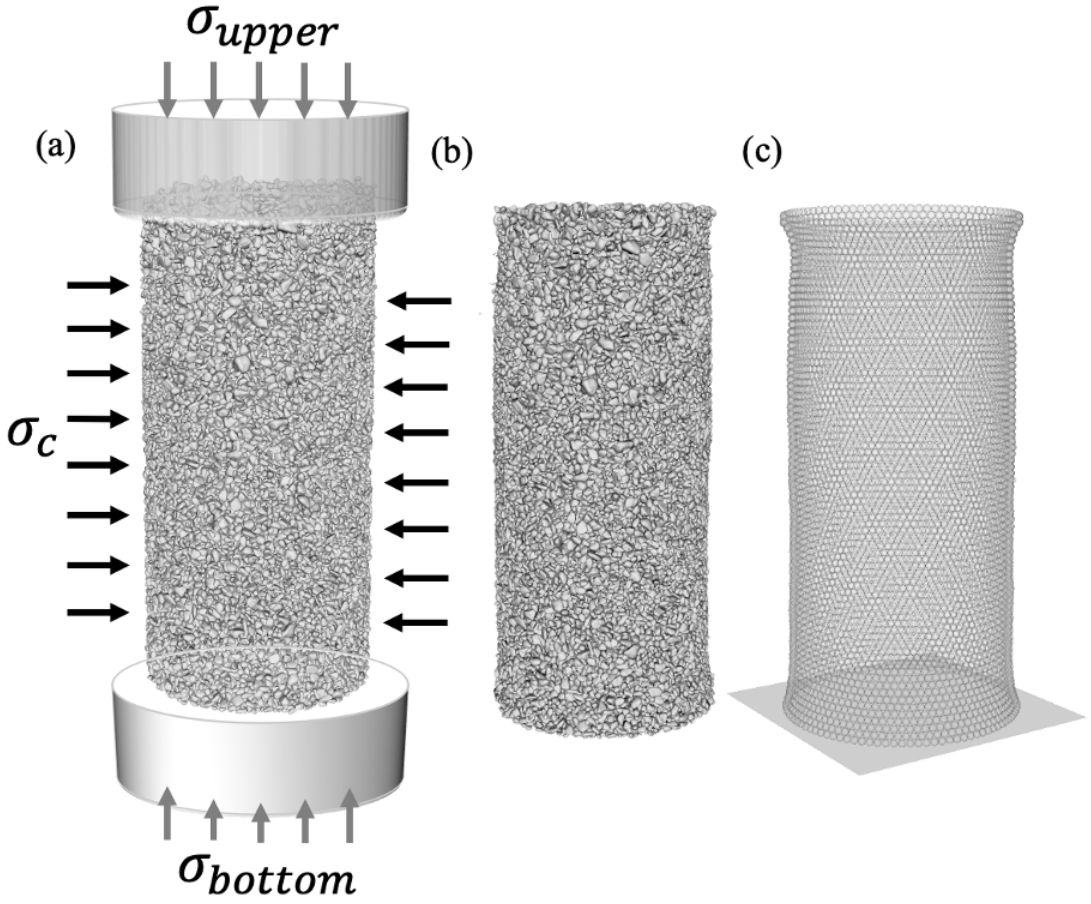

Figure 4.1: Numerical illustration of isotropically consolidation stage (a) before consolidation; (b) after consolidation; (c) membrane deformation.

observations. Then, axial loading was applied following a strain-controlled scheme with a sufficiently low loading rate to maintain the quasi-static condition. The loading was performed in a non-gravity environment to avoid the effect of gravity-induced inhomogeneity.

4.3.2 Scaling a Modeling Representation

LS reconstructed avatars for granular materials produced from XRCT images, numerically recreate the volume, the moment of inertia, and the mass center of the particles. Therefore, a scaling factor is necessary to link numerical models to the laboratory triaxial tests. The value of scaling factor is the XRCT image resolution and varies as per the instrument, typically between $3\mu m$/pixel and $30\mu m$/pixel. A smaller value represents a more accurate capture of the grains shapes and fabric while also inferring a higher numerical cost for both reconstruction and simulation. High-resolution avatars have more discretized nodes

to represent their geometry and to participate force resolution. Indeed, increasing the number of nodes on the avatar surface increases computational effort because contact detection and force resolution are conducted at node level, even though parallel like domain decomposition aids in reducing search complexity and optimizing work balance. We can use either meters or pixels as the length unit. Accordingly, other parameters are converted to their equivalents and are listed in Table 4.1. While the virtual experiment appears to be independent of the scaling factor, as long as the units are consistent throughout the modeling, and the scaling factor can be eliminated from the governing equations. However, the confining pressure must be correctly scaled because it is associated with the mechanical equilibrium of the membrane sphere under confining pressure and grain resistance where the scaling factor does not cancel out.

	Unit in experiments	Unit in numerical models
Density	ρ (kg/m^3)	$\tilde{\rho} = \rho \cdot k^3$ (kg/pixel^3)
Stiffness	s $(N/m = kg/s^2)$	$\tilde{s} = s$ (kg/s^2)
Pressure	P $(N/m^2 = kg/m \cdot s^2)$	$\tilde{P} = P \cdot k$ $(kg/m \cdot \text{pixel}^2)$
Global Damping	ξ $(1/T)$	$\tilde{\xi} = \xi$ $(1/T)$
Friction Coefficient	μ (1)	$\tilde{\mu} = \mu$ (1)
Length	L (m)	$\tilde{L} = L \cdot k$ (pixel)

Table 4.1: Unit conversion with scaling factor k (μm/pixel)

4.3.3 Model Parameters

Two specimens were generated from XRCT images on two different samples taken from the same site but with different image resolutions of $4.3\mu m$/voxel and $9 \sim 10\mu m$/voxel, respectively. The XRCT images with the lower were less clear and produced less angular avatar shapes and eroded grain morphologies. Thus, while the high-resolution scan was able to preserve the in-situ void ratio of the sample, the low-resolution scan failed to capture significant amount of surface roughness, inter-grain contact and more importantly, small grains. As a result, the void ratio of the reconstruction from the low-resolution scan was as high as two, in that avatars were left unsupported by neighbors. Notably, very tiny avatars are often omitted from simulation to guarantee a larger timestep, but the total volume is negligible.

Figure 4.2 shows the two different samples that were reconstructed from the scans. We used the smaller, high-resolution scan for the parametric studies in majority of the simulations, with the sample reconstructed from the low-resolution scans used to illustrate the importance of accurately reconstructing the original fabric of the sand. The microscopic parameters used for LS-DEM investigation of triaxial compression test is tabulated in Table 4.2.

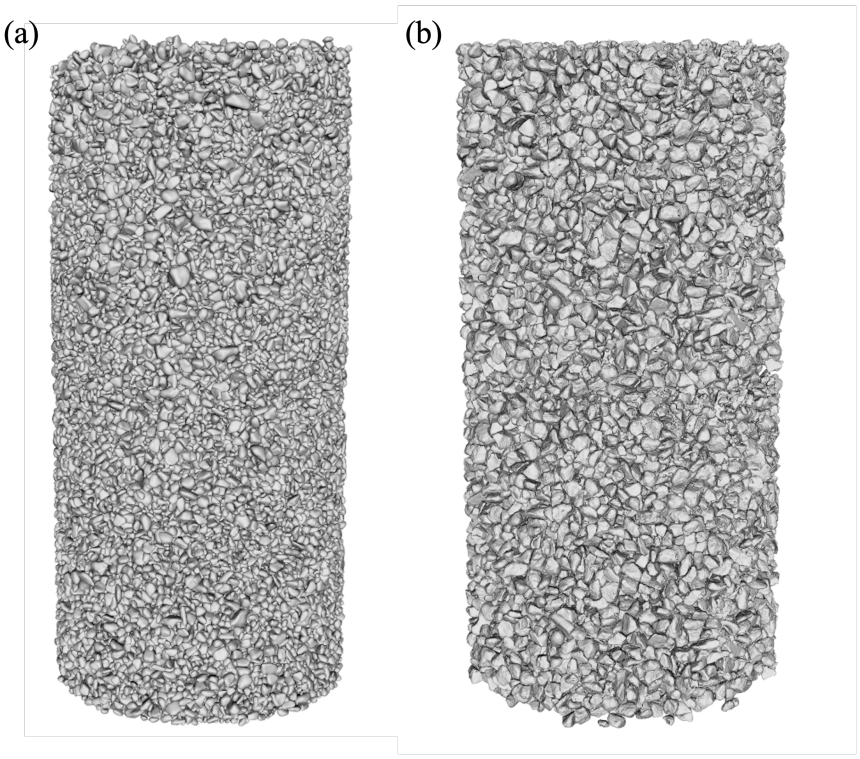

Figure 4.2: Numerically assembly reconstruction from (a) low-resolution ($\sim 70,000$ grains, $9 \sim 10\mu m$/voxel); (b) high-resolution ($\sim 17,000$ grains, $4.3\mu m$/voxel).

4.4 Numerical Modeling Considerations

4.4.1 Numerical Integration Scheme

The penalty-based DEM calculates the resultant force acting on objects and then updates accelerations and velocities using a numerical integrator. It is a second-order accurate finite difference scheme in time-centered form Walton and Braun (1993), which takes into account the 'tangent' change in velocity.

$$\mathbf{v}^{t+\frac{1}{2}} = \mathbf{v}^{t-\frac{1}{2}} + \dot{\mathbf{v}}^t \Delta t \tag{4.16}$$

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \mathbf{v}^{t+\frac{1}{2}} \Delta t \tag{4.17}$$

Which is equivalent to Lee and Hashash (2015):

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \dot{\mathbf{x}}^t \Delta + \frac{1}{2} \ddot{\mathbf{x}}^t \Delta t^2 \tag{4.18}$$

To guarantee the numerical stability, the central finite difference schemes adopt a time step that has to be equal to or less than a critical time step Δt_{cr} because this scheme is

Parameters	Symbol	Values	Units
Global damping	ξ	1.0	$1/s$
Platen			
Platen Radius	R_p	500	pixel
Platen Height	H_p	800	pixel
Density	ρ_p	$2,500$	kg/m^3
Normal stiffness	K_{np}	3×10^4	N/m
Shear stiffness	K_{sp}	2.7×10^4	N/m
Grain-platen friction coefficient	μ_p	0.5	
Membrane			
Sphere radius	r_b	10	pixel
Sphere-sphere normal stiffness	K_{nbb}	50	N/m
Sphere-sphere tangential stiffness	K_{sbb}	50	N/m
Grain-membrane normal stiffness	K_{nb}	3×10^4	N/m
Grain			
Density	ρ_p	$2,500$	kg/m^3
Inter-grain normal stiffness	K_n	3×10^4	N/m
Inter-grain tangential stiffness	K_s	2.7×10^4	N/m
Friction Coefficient	μ	$0.60 \sim 0.75$	

Table 4.2: Microparameters used in LS-DEM of triaxial compression test

conditionally stable. The calculation logic of time step for explicit DEM is based on Newton's second law. To ensure the solution produced by the model remains stable, the time step in each calculation cycle should not exceed a critical time step that is related to the stiffnesses, densities, geometries of modeled objects and minimum eigenperiod of the granular assembly, and often expensive to compute. Instead, Itasca (1998) estimates the critical time step for an assembly of grains using an equivalent single degree of freedom system and concludes that the critical time step is governed by the minimum grain mass $m_{\min}$ and the maximum spring stiffness K_n in an assembly.

$$\Delta t_{cr} \ll \sqrt{\frac{m_{\min}}{K_n}} \tag{4.19}$$

Penzien (1993) and O'Sullivan and Bray (2004) further performed a series of extensive numerical experiments on the appropriate selection of Δt and demonstrate that a much smaller Δt should be used than Δt_{cr} to ensure the stability for two reasons: 1) insufficient small-time step induces significant oscillations in the numerical solution; and 2) unreasonably excessive energy will quickly accumulate with simulation time because of the numerical error associated with the oscillations. For those reasons, a factor of safety of $0.1 \sim 0.2$ is often used. In many granular material simulations, Δt_{cr} may be very small because the particles are assumed to be rigid. Therefore, very high contact stiffness K_n is required to prevent

particle penetration. This also leads to a large number of iterations for simulation, which is fatal in terms of demand on computational resources.

To simulate the triaxial test, the grain assemblies were vertically compressed using a constant downward displacement of the top platen while maintaining a constant confining pressure on the flexible membrane. To achieve quasi-static shearing, the shear rate must be slow enough that the kinetic energy generated by the shear is negligible. As LS-DEM is more computationally intensive than conventional penalty-based methods, we can complete only about $20,000$ iterations per day for a full-scale reconstructed specimen containing on the order of 10^6 grains.

In this case, we wanted our simulation to finish in a reasonable amount of time, which means numerically compressing the specimen to a sufficiently large deformation to observe a fully developed shear band. As a result, the critical time step Δt_{cr} and the maximum number of iterations were related via the loading rate: the amount by which the upper platen is lowered in one step. The velocities of the loading platens are described in units of m/s and m/step in the literature, but seldom reported in relation to the time step for describing the loading strain rate in the compression tests. For example, the recommended loading rate for compression test in the PFC user manual is 0.02 m/s (Itasca, 2004) so that inertial effects, such as platen loads in excess of their quasi-static equilibrium values are avoided. In other studies, a variety of different loading rates, ranging from 0.0016 to 0.3 m/s, were used when modeling virtual quasi-static tests on granular material (Hazzard et al., 2002; Potyondy & Cundall, 2004; Cho et al., 2007; Fakhimi & Villegas, 2007; Park & Song, 2009). These correspond to very high loading rates in the physical world. Intuitively, if the upper platen moves much faster than the grains in direct contact with the platen, responses in the upper portion of the specimen are not adequately transmitted downward, triggering dynamic effects. Our simulation was conducted in a quasi-static conditions and there are two widely used and comparable criteria for choosing a strain rate for simulation. Zhao and Zhao (2019) recommend that a loading rate less than 1% axial strain per second should be sufficiently small to numerically reproduce a quasi-static environment. Alternatively, the inertial number I_{inertia}, as MiDi (2004) and Da Cruz et al. (2005) defined below is used to provide quantitative insights into the selection of loading rate:

$$I_{\text{inertia}} = \dot{\varepsilon}\frac{d}{\sqrt{\boldsymbol{\sigma}_c/\rho}} < 10^{-3} \tag{4.20}$$

Where d is the average equivalent grain diameter in assembly, ρ is the density of grain ($2,650\ kg/m^3$), and $\boldsymbol{\sigma}_c$ is the confining pressure (100kPa), $\dot{\varepsilon}$ is the loading strain rate. The physical meaning of inertia number is the ratio of time scale of grain rearrangement to the time scale of packing shear. MiDi (2004) and Da Cruz et al. (2005) suggested that: although a real triaxial compression test requires I_{inertia} to be as small as 10^{-9}, 10^{-3} is the threshold at which further decreasing I_{inertia} has no effect on the measured macro-properties (Potyondy & Cundall, 2004), and Hazzard et al. (2002) demonstrate that loading velocity has little effect on the mechanical behavior of the sample being modeled as long as the velocity is low

enough to avoid the generation of transient waves.

Also, the formulation of I_{inertia}, $\frac{d}{\sqrt{\sigma_c/\rho}}$ is independent of the scaling factor k after considering the conversion relationship tabulated in previous section, while the strain loading rate $\dot{\varepsilon}$ is dependent on the timestep Δt which is associated with the scaling factor k. Using a larger k results in a larger timestep and hence smaller I_{inertia}. As a result, using a scaling factor k larger than the actual magnitude reduces computational effort and was adopted by Thornton (2000) and Ng (2006). In our work, we explored the possibility of scaling up the grain mass for a larger time step to bring the virtual triaxial simulation as close as possible to that used in the laboratory. To achieve this, we increased the scaling factor to the point where 20% axial strain can be achieved about $40,000$ iterations and the loading strain rate satisfies both practical and quantitative criteria to guarantee a quasi-static condition. It is worth noting that Lee et al. (2012) used a slightly more complicated scheme to increase loading rate while maintaining quasi-static conditions; they discretely moved the top platen at a much higher loading rate and allowed the grains to re-equilibrate due to the top platen's excessive penetration into nearby grains. Alternatively, re-formulating the penalty-based method to an impulse form is another effective way to increase the critical time step (Mirtich & Canny, 1995). In contrast to the penalty-based method, the impulse-based method does not require the use of arbitrarily large stiffness and damping parameters as it solves for velocities directly using Newton's second law's impulse-momentum form. This enables an impulse-based method to use a time step several orders larger than a penalty-based DEM, resulting in comparable computational efficiency. We implemented and evaluated the impulse-based LS-DEM code in other studies discussed in Chapter 5. The primary limitation of the impulse-based method remains its inability to model a system of irregular, nonconvex, non-uniform objects, particularly in a highly confined quasi-static environment in which objects of very different shapes and sizes interact at numerous contact points and have low velocities. This makes collisions within a contact group extremely difficult to solve using a velocity-based algorithm.

As previously stated, the scaling factor k can be increased to allow for a larger time step in order to prolong the time can be modeled. This measure, however, has an effect on the maximum confining pressure applied to the membrane, as it results in excessive overlap between the membrane and the specimen. Similar issues were also reported by De Bono et al. (2012). Thus, increasing the scaling factor requires increasing both the grain-membrane and inter-grain stiffness, or decreasing the magnitude of confining pressure. We conducted a series of simulations to analyze the influence of confining pressure while maintaining a constant confining pressure to contact stiffness ratio. This allowed us to explore how the scaling factor affects the simulation results, as setting the scaling factor to its true value is computationally very intensive. The true scaling factor is in the order of 10^{-5} and the contact stiffness is in the order of 10^5, in order to maintain equilibrium against a 100kPa confining pressure. When the scaling factor is increased from 10^{-5} to 10^{-2}, either the contact stiffness should be increased proportionately, or the confining pressure should be decreased to retain the force balance of membrane spheres. In the simulations illustrated in Figure 4.3,

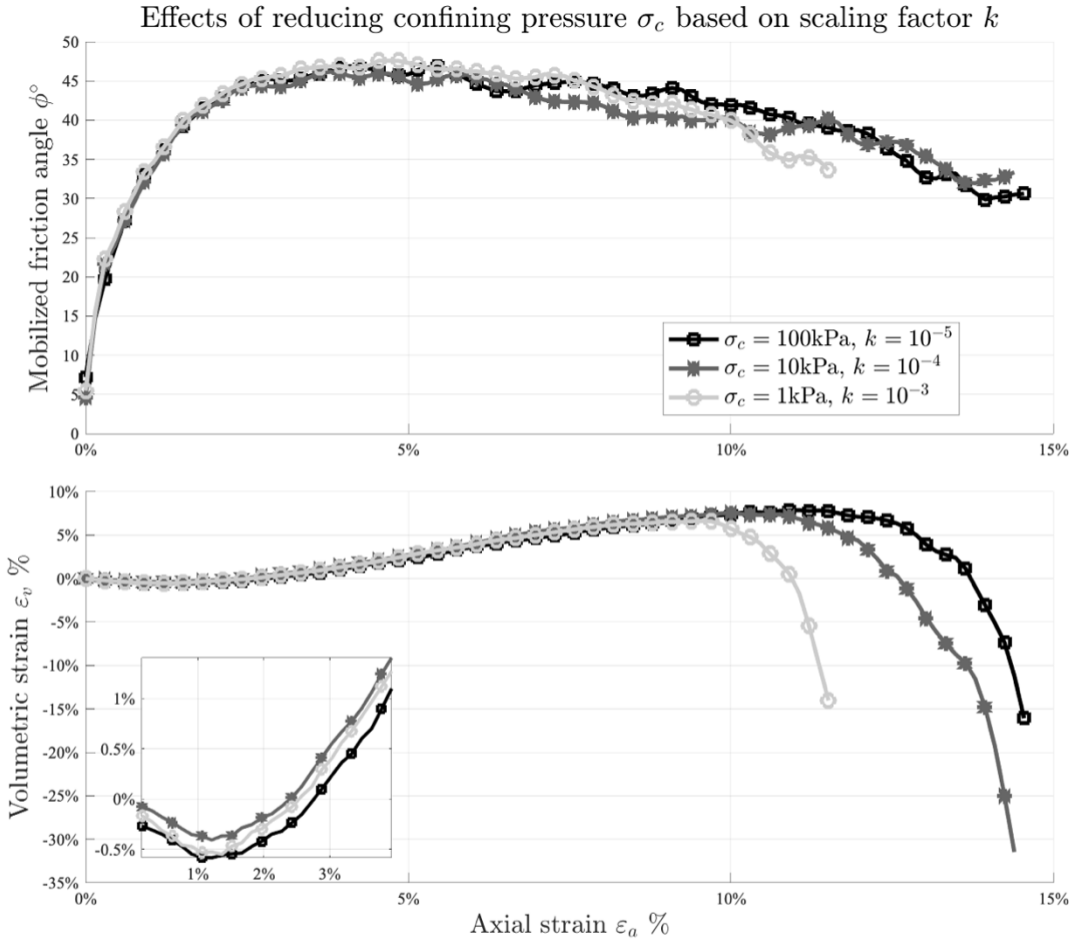

Figure 4.3: The effect of reducing confining pressure due to change of scaling factor.

the confining pressure $\boldsymbol{\sigma}_c$ of 1, 10, and 100kPa corresponds to using 100, 10, and 1 times larger timestep, respectively. There are no obvious differences in deviatoric stress, peak mobilized friction angle, or dilatancy between the three sets of parameters, although there are some minor fluctuations as the specimen approaches the critical state. However, this is of less interest for our current study, and it is believed that this is partly attributable to inaccuracy in volume integration due to the severe membrane distortion at axial strains in excess of 10%. All tests conformed with the inertia number requirement ($< 10^{-3}$) to retain the quasi-static condition.

4.4.2 Flexible Membrane Modeling

For the sake of simplicity, the flexible membrane in our simulated triaxial tests is frictionless, and the degrees of freedom for each membrane sphere are reduced from six to three, obviating the need to compute moment and rotation. This assumption contributes to the model membrane's stability, as numerically integrating Euler's rotation equation is a source of instability. Another problem occurs when the membrane stretches, because a few grains contained within can escape and become uncontrollable. Although this has no effect on the simulation results, it invalidates the subroutine used for dynamic domain re-decomposition. Specifically, the purpose of domain re-decomposition is to re-generate bins, blocks and subdomains based on the current grain distribution to best balance workload across processors. When a grain escapes from the flexible membrane and flies far away from the main body, the re-decomposition strategy will incorrectly consider a much larger domain as the current configuration and distribute grains accordingly, resulting in the majority of processors being idle and one processor performing all work. To address this issue, we used a double-layer membrane with a slightly larger sphere radius in the outer layer. This resulted in membrane spheres being displaced slightly to cover gaps in the first layer, preventing grain from escaping. The confining pressure was then applied to both layers with 80% of the pressure on the inner layer and 20% on the second layer to obtain the desired magnitude.

The computational cost of the membrane is determined by the radius ratio of the grain to the membrane sphere. Larger elements are more computationally affordable and more numerically stable, making it better able to withstand massive deformations when spheres are pushed into an incorrect geometry. However, if the ratio decreases by using larger membrane spheres, the potential for leakage of small soil grains through the membrane increases. Due to volumetric dilation caused by large deformation, the pore size (spacing between membrane spheres) can also increase. As a result, an appropriate radius ratio should be chosen to strike a balance between computational time and grain leakage. Kawamoto et al. (2018) determined the membrane sphere size as one third of the average size in the specimen and Kim et al. (2020) selected the value to be as twice the radius of the smallest grain. The effect of membrane sphere radius was investigated for an assembly with an average equivalent radius of 18, the size of sphere is 6, 8 and 10 for the first layer of flexible membrane and 1.1 times larger for the second layer, no grain leakage was detected for all tests. The simulation results in Figure 4.4 for $r_b = 8$ and $r_b = 10$ matched, whereas the simulation with $r_b = 6$ suffered numerical instability at high shear strain and manifested itself by uncontrolled volumetric contraction. The stability analysis of membrane element is analogous to the grain assemblage stability in that both can be thought as a mass-damped system

$$m\ddot{\mathbf{x}} + \xi\dot{\mathbf{x}} + k\mathbf{x} = \mathbf{0} \tag{4.21}$$

The contact model between membrane spheres can then be calibrated to match the behavior of the particle-based membrane to that of a real membrane at the selected radius ratio. The effect of membrane stiffness K_m on the macroscopic behavior of a granular assembly was investigated using a series of parameter studies ranging from $50N/m$ to $1,000N/m$, as

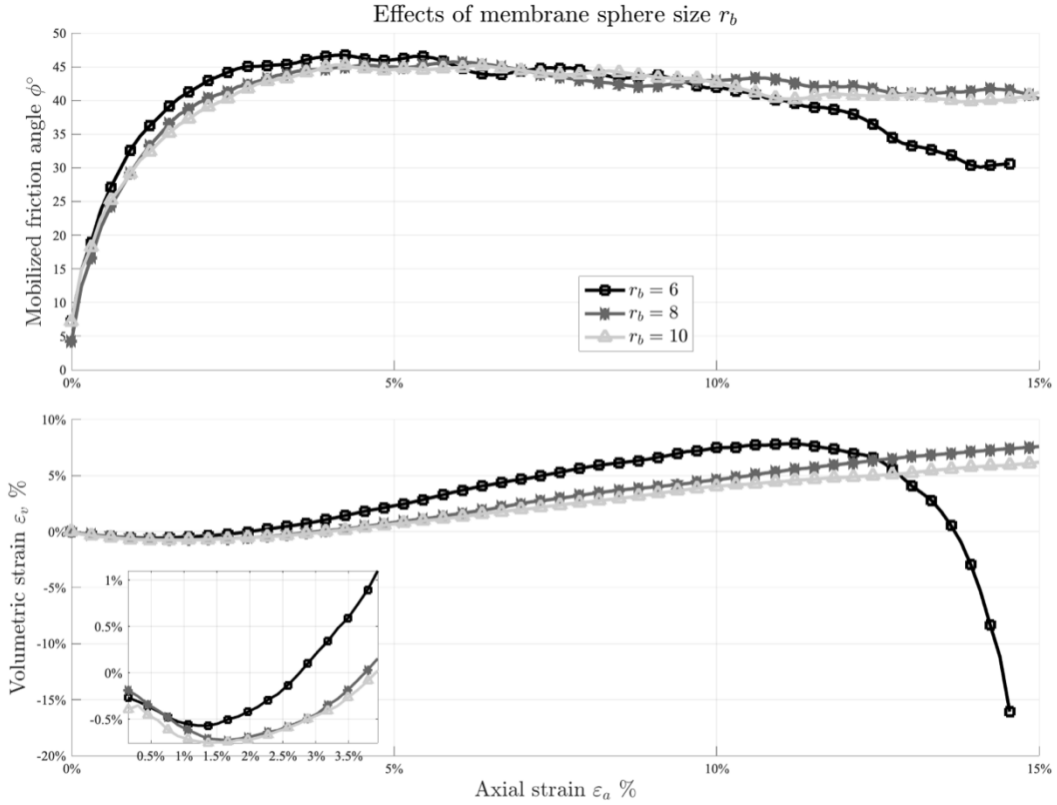

Figure 4.4: The effect of membrane sphere's radius.

shown in Figure 4.5. The results show that the membrane stiffness does not have significant influence on peak mobilized friction angle, or initial dilatancy in a triaxial simulation. While a stiffer membrane provided a stronger support for the specimen, as indicated by a slightly larger elastic modulus and less volumetric contraction. By making the boundary flexible, the physical integrity of specimen is maintained while its tolerance for large deformations is increased. In addition, the system gains additional degrees of freedom to deform and more possibilities for shear band development.

4.4.3 Numerical Stability of Angular Velocity Integration

A LS avatar integrates angular velocity and rotation using Euler's rotation equation, which is not necessary for conventional DEM simulations of spheres or discs.

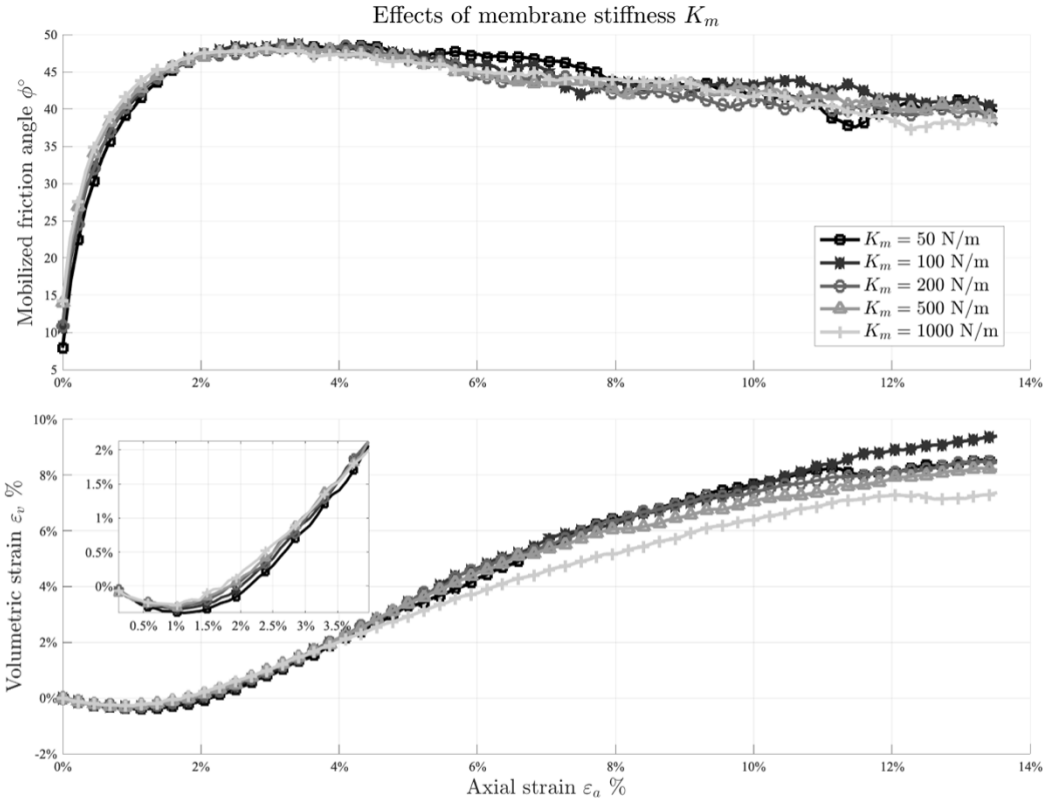

Figure 4.5: The effect of membrane stiffness K_m.

$$\begin{aligned} \alpha_1 &= [M_1 + \omega_2\omega_3(I_2 - I_3) - \xi I_1\omega_1]/I_1 \\ \alpha_2 &= [M_2 + \omega_3\omega_1(I_3 - I_1) - \xi I_2\omega_2]/I_2 \\ \alpha_3 &= [M_3 + \omega_1\omega_2(I_1 - I_2) - \xi I_3\omega_1]/I_3 \end{aligned} \tag{4.22}$$

Where α_i is the angular acceleration, ω_i is the angular velocity, I_i is the (diagonal) moment of inertial tensor in the principal body-fixed frame, and M_i is the torque vector in the principal body-fixed frame.

Euler's equations are nonlinear and implicit due to the presence of angular velocities on both sides. As a result, a predictor-corrector algorithm is proposed for appropriately integrating the rotational components of motion; this method resembles a fixed-point method and is stable and convergent when the timestep is sufficiently small. Since LS-DEM employs a time step smaller than the critical time step, this value also ensures the predictor-corrector algorithm's numerical stability. Additionally, global damping is used to dissipate excessive energy, and to further reduce the possibility of accumulated numerical errors. However,

simulations can still occasionally explode, and it is hypothesized that this is due to some grains erroneously attract forces and moments that are too large and cause the predictor-corrector scheme explodes within a single iteration.

The principal reason for such event in our simulations is that in constructing the avatars from images of grain assemblies, grains can be accidentally reconstructed inside or cross penetrating other grains. We addressed this issue by limiting the coordination number of a grain, which is correlated with the packing density of a granular assembly. The simplest definition of the coordination number is the mean number of contacts per particles:

$$Z = \sum_{i \in N} \frac{N_c^i}{N} \tag{4.23}$$

Where N_c^i is the total number of contacts of the i-th grain and N the total number of grains in the assembly. The scalar coordination number is strongly related to the internal fabric structure of the granular material and is highly correlated with the mechanical stability (Nouguier-Lehon et al., 2003; Mitchell & Soga, 2005). In addition, the coordination number is sensitive to grain shape, which has a significant impact on deformation of a granular assembly (large drop in mean coordination number corresponding to larger dilatancy). Irregularly shaped grains result in greater coordination number. The experimental data indicates that a grain coordination in naturally deposited sands is between 8 and 16, and our parametric study shown in Figure 4.6 demonstrates that as long as the threshold value is within a reasonable range, no significant differences in the simulation results are observed. Furthermore, increasing the threshold value to 64 did not incur numerical instability. This confirms that numerical instability is primarily due to the severe overlap between avatars and hence scaling extremely large forces and moments down is a rational solution. Interestingly, reducing the coordination number to four seems to have very little impact on the simulation results. This is because the external loading consists primarily of confining pressure from the encased flexible membrane and compression forces from the downward moving platen; this leads to only a small portion of the assembly being directly subjected to those forces.

4.5 Parametric Study and Model Calibration

The objective of parameter study was to determine the suitable micro-parameters for DEM simulations. The influence of membrane properties and scaling factor were already discussed. The other model parameters were determined by trial and error until the simulation produced an initial stiffness and peak friction angle that were comparable to those obtained from a triaxial compression test on a naturally deposited sand specimen. All experiments were conducted on an assembly of $4,852$ avatars constructed from high-resolution ($4.3\mu m$/voxel), excellent-quality images such that original soil fabric was well-preserved and was found to behave consistently with real laboratory results. The number of avatars is sufficient for study of macroscopic behavior and the computational cost is affordable for parameter studies.

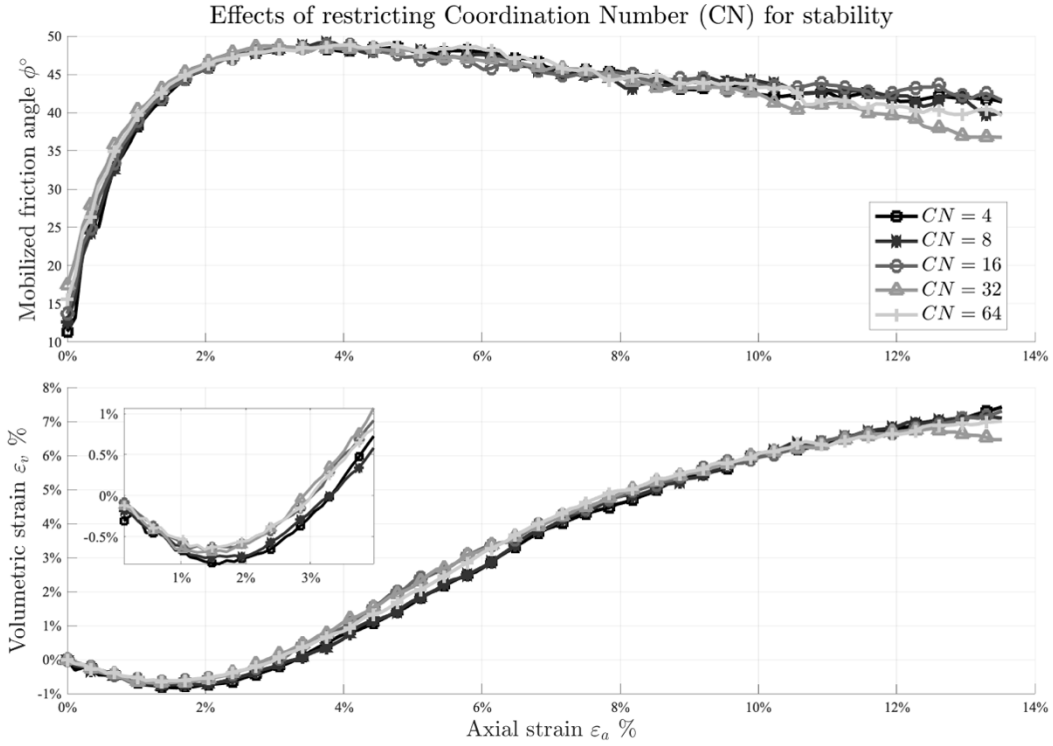

Figure 4.6: The effect of restricting coordination number for numerical stability concerns.

4.5.1 LS Avatar Node Density

Grain avatars are characterized and represented by the discretized nodes seeded on the surface and the node density is closely associated with the computational cost in inter-grain force resolution. This representation is similar to a triangulated surface mesh except that LS-DEM does not store connectivity information between nodes and therefore does not consider edge-surface collision. The density of nodes on a grain is entirely up to the designer. It has no effect on the representing geometry but does have an effect on the computational complexity associated with force resolution.

We studied the consequences of decreasing the node density of the avatar while maintaining a balance between precise grain morphology and simulation time. To do this, we considered only the N-th nodes and skipped the rest during the force resolution step when iterating over discretized nodes on the master avatar's surface and marked down the speed-ups. To produce similar magnitudes among different runs, we multiply the resulting forces, moments, and coordination number by N. The results in Figure 4.7 indicate that the degree of reconstruction fidelity significantly affects the entire computation time, since decreasing node density results in linear performance benefits. The initial stiffness and maximal mobi-

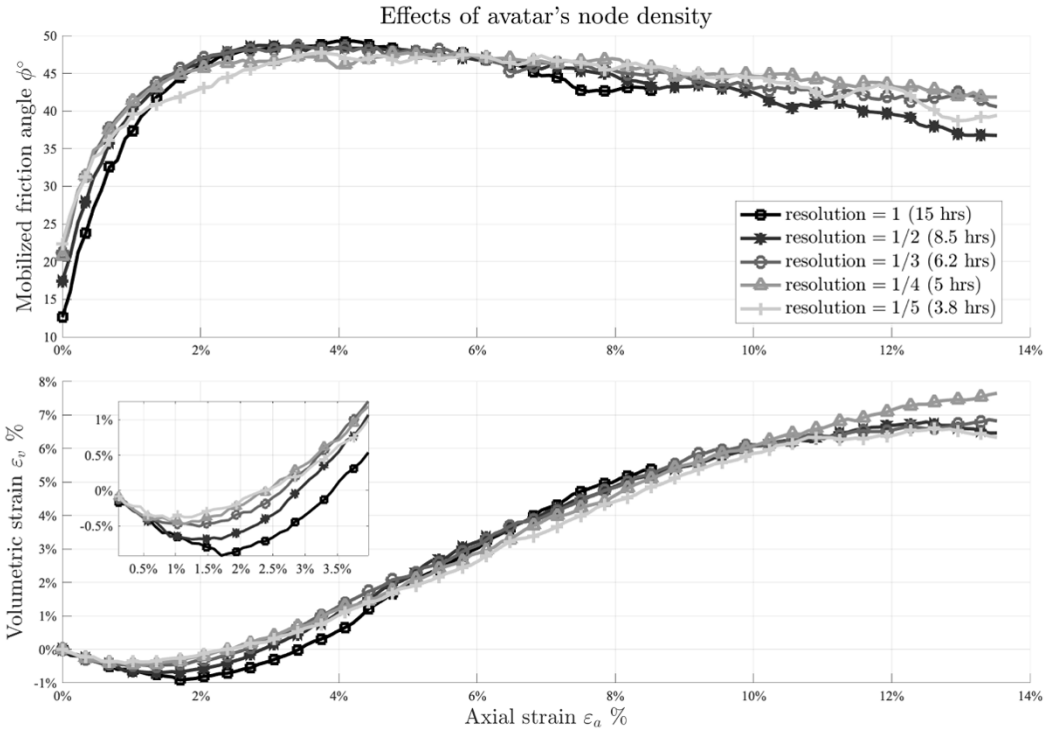

Figure 4.7: The effect of avatar node density.

lized friction angle are reasonably consistent across simulations with $N \leq 3$. Despite the fact that the amount of dilatancy is reduced even for scenario $N = 2$, all simulations achieved the same critical condition. This data indicates that we can decrease the resolution of the avatar for faster code execution but we do undertake the risk of missing contacts if too many nodes are skipped. While we believe that reducing node density on reconstructed avatars is a viable acceleration strategy, we are still in need of clean, high-resolution XRCT images for high-fidelity grain morphology reconstruction, as the two are fundamentally different aspects. We explored herein the implications of node density with the underlying premise that grain geometry is correctly represented and that small grains are retained to accurately duplicate the void ratio of the overall assemblage. By comparison, if the underlying dataset (XRCT pictures) is significantly contaminated, we are incapable of reproducing naturally deposited soil fabric in-situ, let alone modeling mechanical and kinetic behaviors on the numerical equivalents.

4.5.2 Confining Pressure

The magnitude of confining pressure in our virtual numerical modeling depends on the membrane-grain contact stiffness as too large confining pressure will push membrane spheres completely enter into grains, leading to heavily distorted membrane and uncontrolled behavior afterwards. Thus, the ability to withstand high confining pressure indicates that the assembly is well-packed, as membrane penetration effect will result when the packing is loose and when the membrane element is small. The range of confining pressure can be estimated from the force equilibrium relationship of membrane spheres and the maximum value is found around 500kPa with the membrane-grain stiffness is fixed to be $30,000$N/m. Which is enough for our purpose of study to investigate soil behavior under non-crushing stress environment where the stress-strain curve did not show the brittle-ductile transition. The results (Figure 4.8) show that the friction response increase with increasing confining pressure, strain softening takes place with all confining pressures and the addition of confinement decreases the post-peak load-bearing capacity of sample, thus making the post-peak curve steeper. When the confining pressure $\boldsymbol{\sigma}_c$ is greater than 200kPa, the initial stiffness, peak mobilized friction angle and volumetric dilation seem to be identical, while a larger confining pressure drives membrane element enter into the specimen and halts the simulation. It is important to note that in this case the membrane configuration controls the maximum confining pressure in the simulation. Alternative membrane construction could be used to model tests with higher confining pressures.

4.5.3 Normal and Shear Stiffness Ratio

Two linear springs are inserted at contact between grains, one for the normal stiffness specified by the contact K_n and another for the shear stiffness specified by K_s. Normal force is proportional to normal stiffness and the overlap of two grains; shear force is proportional to shear stiffness and the relative rotation of two grains. The effect of altering the ratio of normal to shear stiffness K_s/K_n was calibrated using a series of simulations. The friction coefficient has been set to 0.75, while K_s/K_n has been set to 0.2, 0.4, 0.6, and 0.8. Figure 4.9 illustrates the effect of the normal to shear stiffness ratio on the mobilized friction angle ϕ and volumetric strain ε_v vs axial strain ε_a with a confining pressure of 100kPa. Elastic stiffness increases as the normal to shear stiffness ratio increases. Additionally, the peak strength increased somewhat, resulting in a minor rise in the angle of internal friction. On the other hand, the volumetric strain at initial dilatancy is nearly identical for all ratios, however it increases more pronouncedly with the ratio K_s/K_n as the specimens were sheared to approach the critical state. A high ratio of K_s/K_n represents high ability of tangential deformation resistance for grains in contact, and the resemble effect is to increase the lateral deformation for the same axial deformation, generating more stable force network in the axial direction, which makes the sample stiffer. K_s is frequently smaller than K_n in DEM modeling to encourage numerical stability and prevent producing fictitiously large moments that could blow out the simulation.

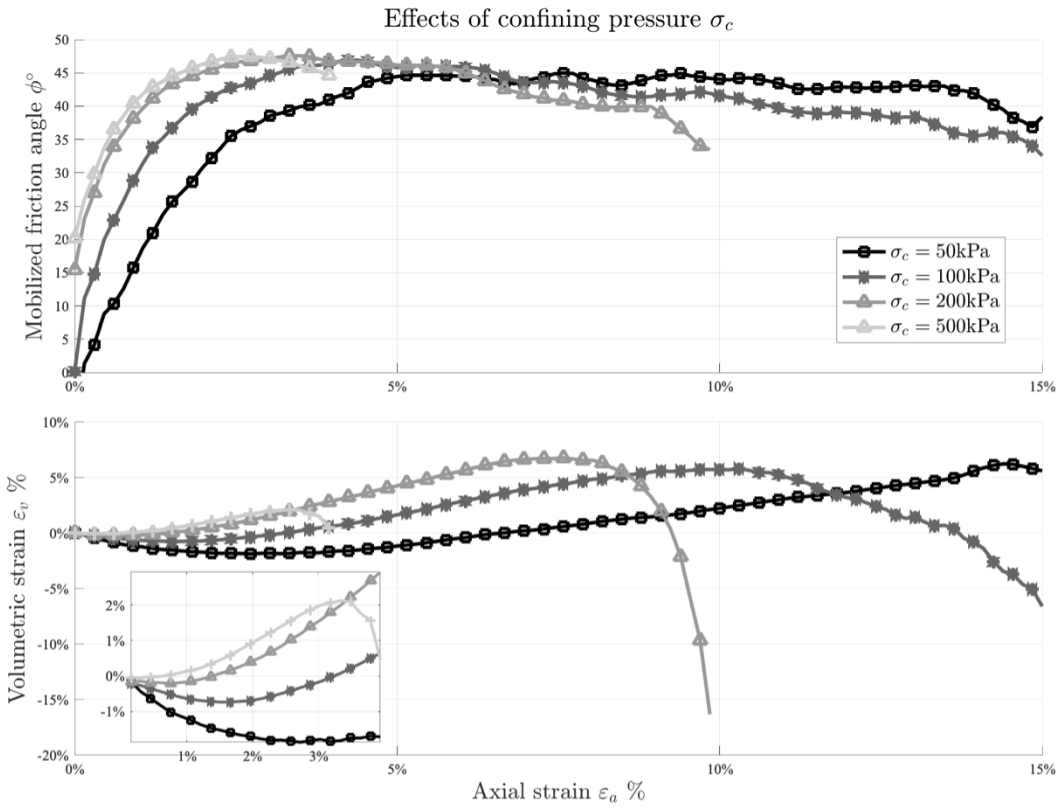

Figure 4.8: The effect of confining pressure, simulations with $\boldsymbol{\sigma}_c$ = 200kPa and $\boldsymbol{\sigma}_c$ = 500kPa ended earlier due to membrane distortion.

4.5.4 Grain Shape

The initial positions of high-resolution reconstructed avatars were employed to generate a spherical counterpart with equivalent radius to ensure the same initial void ratio as the LS reconstructed avatars (Figure 4.10). There may be some exaggerated or even unreasonable intersections for some spheres in the initial packing. Therefore, the packing is further subjected to more DEM cycles in the absence of gravitational force to reach a state of mechanical equilibrium. Throughout the course of stabilization process, spheres' velocities are periodically reset to zero to avoid excessively large velocity due to possible significant penetration and collisions. As summarized by Mitchell and Soga (2005), even though the void ratios of an assemblage of uniform spheres are in the range of 0.35 to 0.91 depends on different packing patterns, the net void ratio may not be much different from the real granular assembly. Because, on the one hand, smaller grains can occupy pore spaces between larger grains, and on the other hand, irregular grain shapes produce a tendency toward lower densities and higher porosities. The void ratio of an assembly of reconstructed avatars is

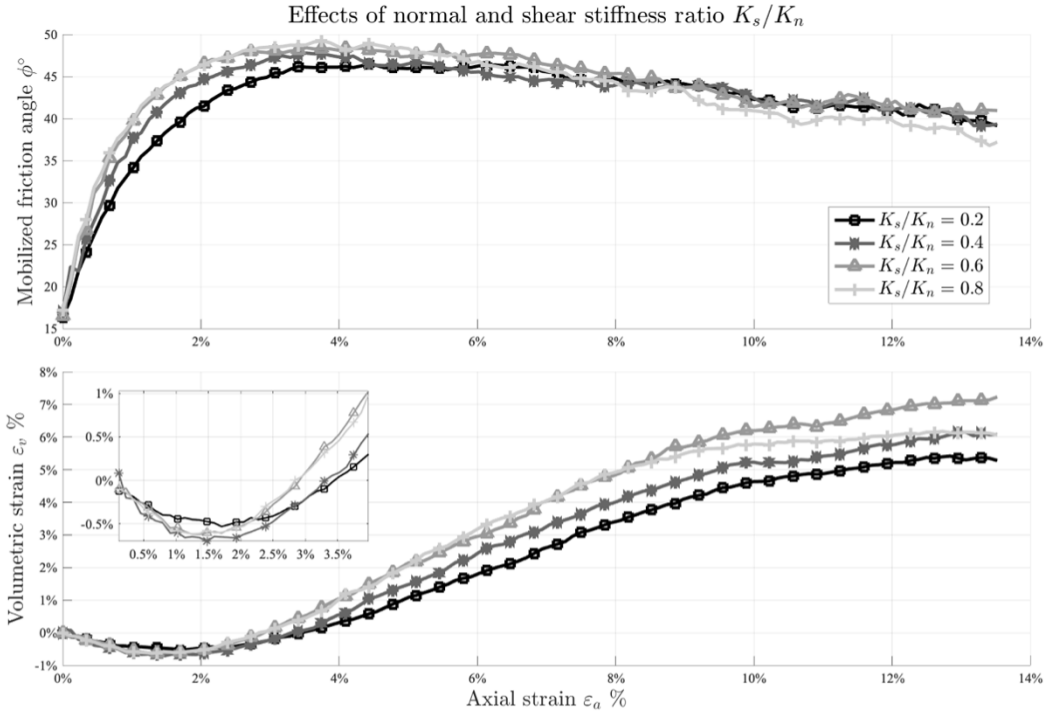

Figure 4.9: The effects of ratio of normal and shear stiffness K_s/K_n.

slightly larger than the true value as tiny grains were not captured during image processing and also eliminated from simulation to ensure numerical stability.

The parameters of modeling idealized spherical particles were not calibrated and the rolling friction and bonding were not considered to compensate the influence of grain shape and surface roughness in current work, this is thought to be the primary reason that spherical particles did not exhibit the peak mobilized friction. The influence of internal friction coefficient is illustrated in Figure 4.11 to show that, regardless the parameter settings, spherical assemblies were not able to display dilatancy and peak mobilized friction angle which due mainly to particle interlocking and surface roughness, even though the void ratio is kept the same as the assembly of avatars. When the internal friction coefficient μ decreases and eventually becomes zero, specimen deformation tends to be a more localized process, as interactions can only be transferred via direct normal contact, and no friction means that spheres have a higher degree of rotation and can find the optimal location to accommodate themselves without influencing others to a greater extent.

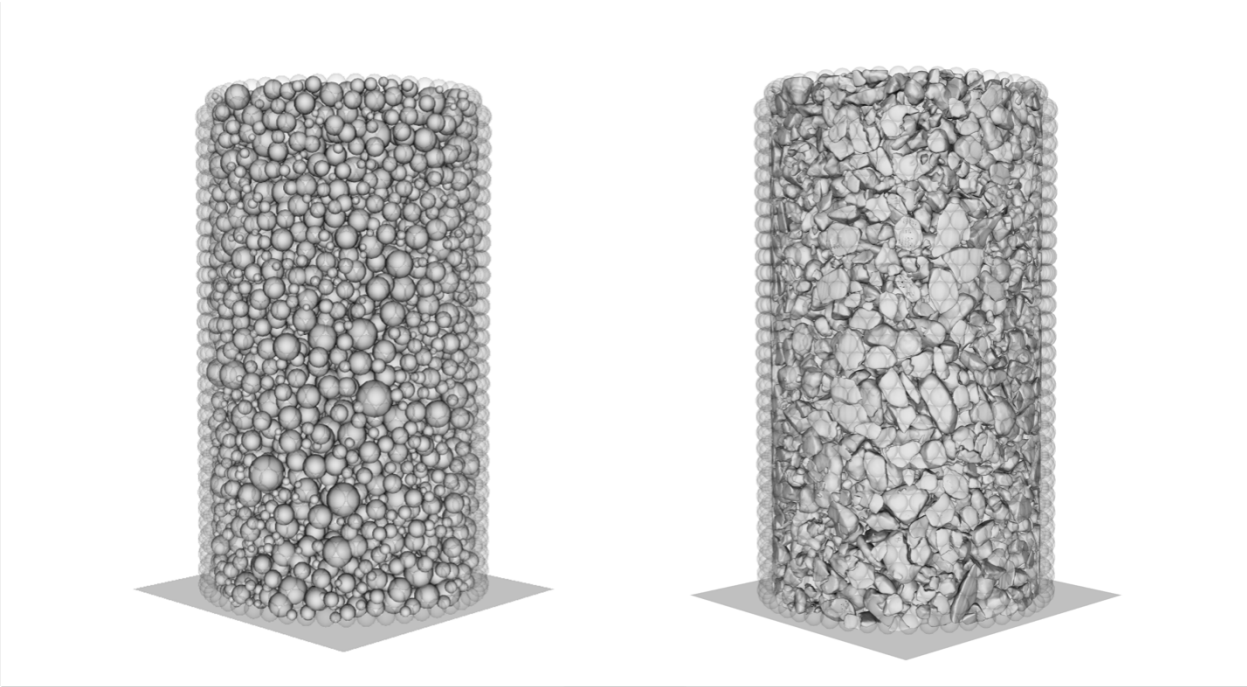

Figure 4.10: Left: An assembly of spherical particles. Right: An assembly of LS reconstructed avatars. Both specimens have identical void ratio.

4.5.5 Initial Void Ratio

The initial void ratio of a specimen is critical to the stability of the simulation because it can be unrealistically high or low and outside the normal range. For example, a numerical specimen that is reconstructed from low-resolution or polluted images has difficulty preserving small grains and capturing sharp contact edges; as a result, the numerical void ratio is unfaithfully large, individual grain is isolated and unlikely to be supported by its neighbors in order to resist confining pressure, causing the membrane to shrink significantly. To solve this issue, the assembly has to settle before performing a virtual compression test, but an erroneous choice of settlement force can result in an inhomogeneous and excessively dense specimen, which means grains may interpenetrate. In such case, very large forces are generated at the initial time step as the grains attempt to adjust their positions to avoid overlapping but are unable to do so without interfering others.

This issue not unique to LS-DEM; in conventional DEM, which represents grain with simpler geometries, stabilizing the specimen prior to simulation is a standard step to avoid numerical instability (Itasca, 2004). But our issue is unique in that we want to preserve as much of the soil fabric as possible and to reconstruct avatars with the same initial positions and rotations as they were in-situ. This is one of the most significant challenges, and there is still a sizable research gap to close. To achieve the desired void ratio when using grain

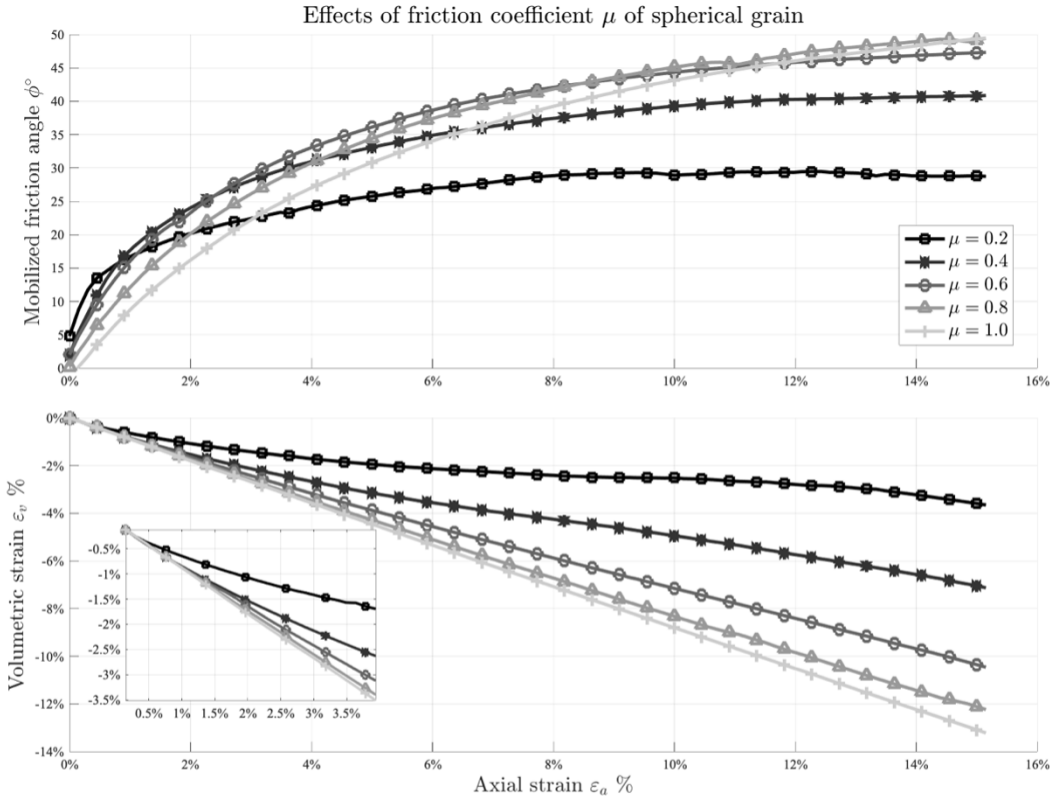

Figure 4.11: The effect of internal friction coefficient of spherical grains.

reconstruction from low resolution scans, we used a numerical pluviation scheme to control the drop height: the loose specimen is divided into several segments along its length, and a segment is activated at a time, subjected to a vertical force, and pluviated from the prescribed height. We encased the sample in a cylinder during pluviation to prevent excessive lateral movement and to avoid disturbing the original fabric. Following that, the packing is further subjected to more DEM cycles without any external forces to reach a state of mechanical equilibrium, this procedure ceases when the internal stress of packing is stabilized at a value that is considerably smaller than the target confining pressure ($\boldsymbol{\sigma_c}$ = 100kPa in current work). However, all of the above-mentioned numerical treatments alter the original fabric. As grains bounced against one another and migrated in random directions during dynamic pluviation to the desired void ratio, their initially ordered arrangement became random, defeating our goal of preserving as much of the original soil fabric as possible. In contrast, these steps are not required when the grains and fabric are constructed using high resolution XRCT images.

The influence of the initial void ratio and particle shape is illustrated in Figure 4.12. The

fabric reconstructed from the high-resolution images could achieve almost the same void ratio as the real sample, although fair number of tiny avatars was omitted from numerical specimen for the sake of numerical ease, the total volume occupied by them was quite small. In contrast, the initial void ratio was significantly less than its true value for the numerical specimen reconstructed from the low-resolution scan and different void ratio can be manufactured via a numerical settlement or pluviation procedure. In addition to the high-resolution specimen and its spherical counterpart mentioned in previous discussion, we constructed another specimen containing $\sim 17,000$ avatars using medium-resolution images ($9 \sim 10\mu m$/voxel). The results show that the relatively dense specimens for both high- and medium-resolution LS avatars show strain hardening, a peak at around $\varepsilon_a = 2.0 \sim 3.5\%$, and gradual softening. Spherical grains and low resolution avatars, on the other hand, exhibit purely strain hardening and volume contraction until they achieve critical states. In comparison to perfectly spherical grains, irregularly shaped avatars have larger mobilized and peak friction angles and are less likely to rotate than spherical grains. The peak mobilized friction angle increases from $\phi_p = 35^\circ$ (spheres) up to $\phi_p = 49^\circ$ (high resolution, high density avatars). In turn, the residual internal friction angle coincides for all simulations. The critical condition is always attained at a considerable axial strain around $\varepsilon_a = 15\%$, where the vertical normal stress remains constant with the specimen deforming at constant volume. This implies that grain shape is not significant in determining the global critical internal friction angle, as would be expected.

4.6 Virtual Triaxial Compression Test

The model reconstructed from the high resolution images ($4.3\mu m$/voxel) contains $\sim 20,000$ avatars with ~ 750 nodes on average for a fully captured grain geometry. The model reconstructed from slightly lower resolution specimen ($9 \sim 10\mu m$/voxel) scans resulted in $\sim 70,000$ substantially less angular avatars with 250 discretized nodes per grain. We then performed numerical triaxial tests assuming a range of material friction coefficient from $\mu = 0.60$ to $\mu = 0.75$, which spans the range of typical values for quartz rich sand with the upper value corresponding to results obtained from the miniature triaxial tests on the undisturbed samples.

The low-resolution specimen was allowed to settle via a pluviation-like process under artificial gravity to re-establish contact between grains before isotropic consolidation. The void ratio was monitored throughout this process as accompanied by several other indicators such as average coordination number and macroscopic stress. Following settlement, a number of iterations were used to alleviate internal stress between grains and to correct excessive inter-grain overlap due to settlement. This step was continued until the internal stress was sufficiently smaller than the confining pressure, 100kpa considered herein. Not surprisingly, those preparation steps did not preserve the in-situ orientation, contact and structure of grains in the original sample. By contrast, the high-resolution specimen did not require either of those steps. Even though the grains were not yet in contact with neighbors, as indicated

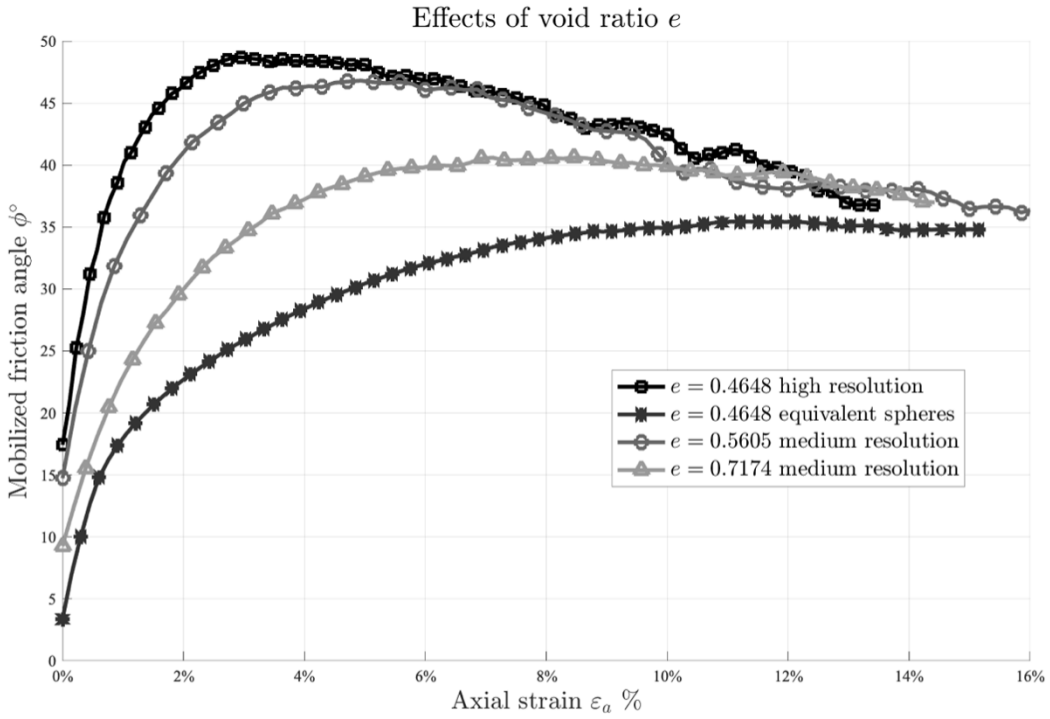

Figure 4.12: The effect of void ratio, considering both the effect of grain shape and the effect of reconstruction fidelity.

by the coordination number at the onset of simulation, due to the fact that avatars were reconstructed one at a time and left only small gaps between them, they were quickly brought together after the application of confining pressure and to a large extent recovered the soil fabric. The high-resolution specimen had initial void ratio $e = 0.60$, which is very close to the experimental data of 0.61. To make the simulations comparable, the low-resolution specimen was also constructed with same initial void ratio $e = 0.59$.

The models were then compressed following a strain-controlled scheme with a sufficiently low loading rate to maintain the quasi-static condition by keeping the inertia number I_{inertia} below 10^{-3} (note that the same loading rate as in the physical experiment is not possible because it would be computational intractable). We monitored several indicator to examine quasi-static condition, making sure that the total kinetic energy of avatars was smaller than 10^{-3} throughout the simulation, and the pressures on both platens was equal to the major principal stress of specimen.

4.6.1 Macroscopic Frictional Behavior

The friction coefficient between grains is a physical characteristic that is difficult to quantify and is strongly dependent on the mineralogy and chemical composition of the grains. Peak stress and mobilized friction angle rise as the inter-grain friction coefficient rises which agrees with other studies like Cui and O'Sullivan (2005), Abriak and Caron (2006) and Hazzar et al. (2020). For high-resolution sample, increasing friction coefficient from $\mu = 0.6$ to $\mu = 0.75$ result in a significant increase in peak deviatoric stress (549kPa to 652kPa) but a gentle increase in peak mobilized friction angle (46.2° to 49.0°). Our experimental results reveal that the critical state angles for naturally deposited sand and reconstituted sand are 38° and 32°, respectively, which corresponds to friction coefficient 0.62 and 0.78. In micro scale, the mineralogy parameters of quartz, the major constitution of sands is reported as 35° by Mitchell and Soga (2005) and is slightly smaller than the computational value we calibrated. In addition, Mitchell and Soga (2005) collected experimental data and suggested that the macroscopic friction angle is nearly independent of interparticle friction angle.

The evolution of the mobilized friction angle and volumetric strain is depicted in Figure 4.13 for low- and high-resolution numerical specimens. The models exhibit markedly different macroscopic behavior, despite having the same initial void ratio and being tested under identical conditions. The low-resolution model exhibits principally strain hardening and reaches a stress plateau. It almost perfectly matches the experimental data on reconstituted sand and supports the hypothesis that pre-processing steps for low-resolution reconstruction significantly altered the original soil fabric. By contrast, the high-resolution model exhibits significant strain softening upon reaching a peak mobilized friction angle, the peak mobilized friction angle agrees well with experimental data ($\phi_p = 51.2°$) on naturally deposited sand. Most importantly, this results demonstrate the importance of the depositional fabric. The reconstituted, pluviated model show strain hardening behavior reaching roughly identical mobilized friction angle between 29° and 30° regardless of the particle friction angle. In contrast, the model that reproduces the depositional fabric with high fidelity, shows a well-defined peak, mobilized friction angle ranging from 46° to 49° over the range of particle friction angles from 30° to 37°.

4.6.2 Evolution of Mean Coordination Number

We further investigated the mechanical responses of these specimen from a microscopic point of view, the emphasis was placed here upon the coordination number which is defined as the number of contact on a grain and recognize that there could be multiple contacts between a pair. Coordination number is one of the most commonly used important parameters for characterizing granular fabric and is found sensitive to particle shape and increases with the increase complexity of grain shape. As displayed in Figure 4.14, the high-resolution specimens with both value of friction coefficient have a higher initial average coordination number, which makes rotation more difficult and results in the formation of more contacts in response to the displace tendency. The coordination number of low-resolution specimen

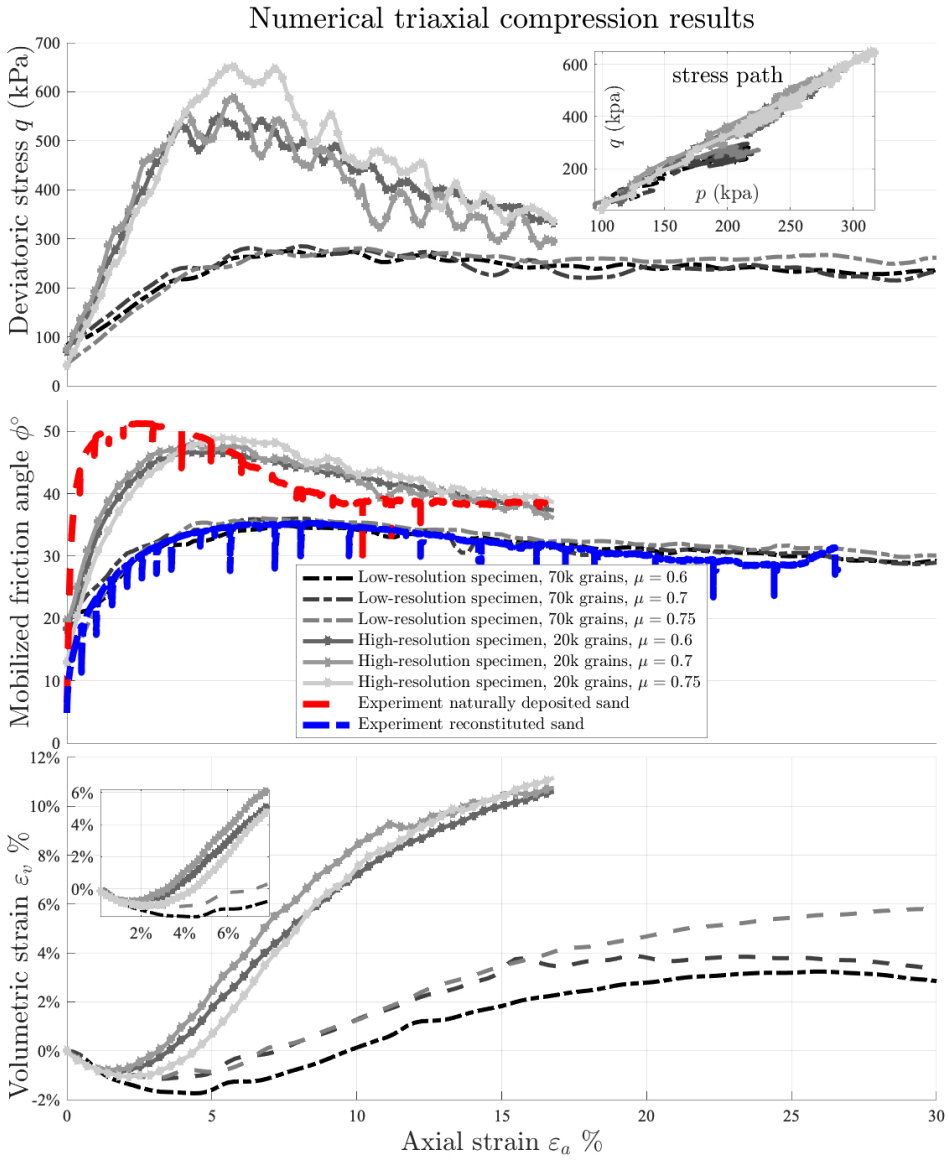

Figure 4.13: Frictional response, i.e., deviatoric stress, mobilized friction angle and volumetric strain for numerically low- and high-resolution specimen in comparison with experimental data.

continue to increase and reach a plateau, implying that newly formed and broken contacts balance each other at high strains and reaching a critical state. In contrast, the change in coordination number for high-resolution specimen is greater for grains due to a rougher surface and a more angular shape than. It also significantly affects granular deformation in that a drop in mean coordination number corresponding to larger dilatancy. In our appli-

cation, the evolution of coordination number is highly associated with the image resolution, and we found the mean coordination number of a high-resolution assembly is almost twice larger than those of the low-resolution reconstruction. This is to some extent reflects the interlocking enhanced by more angularity of high-resolution grains.

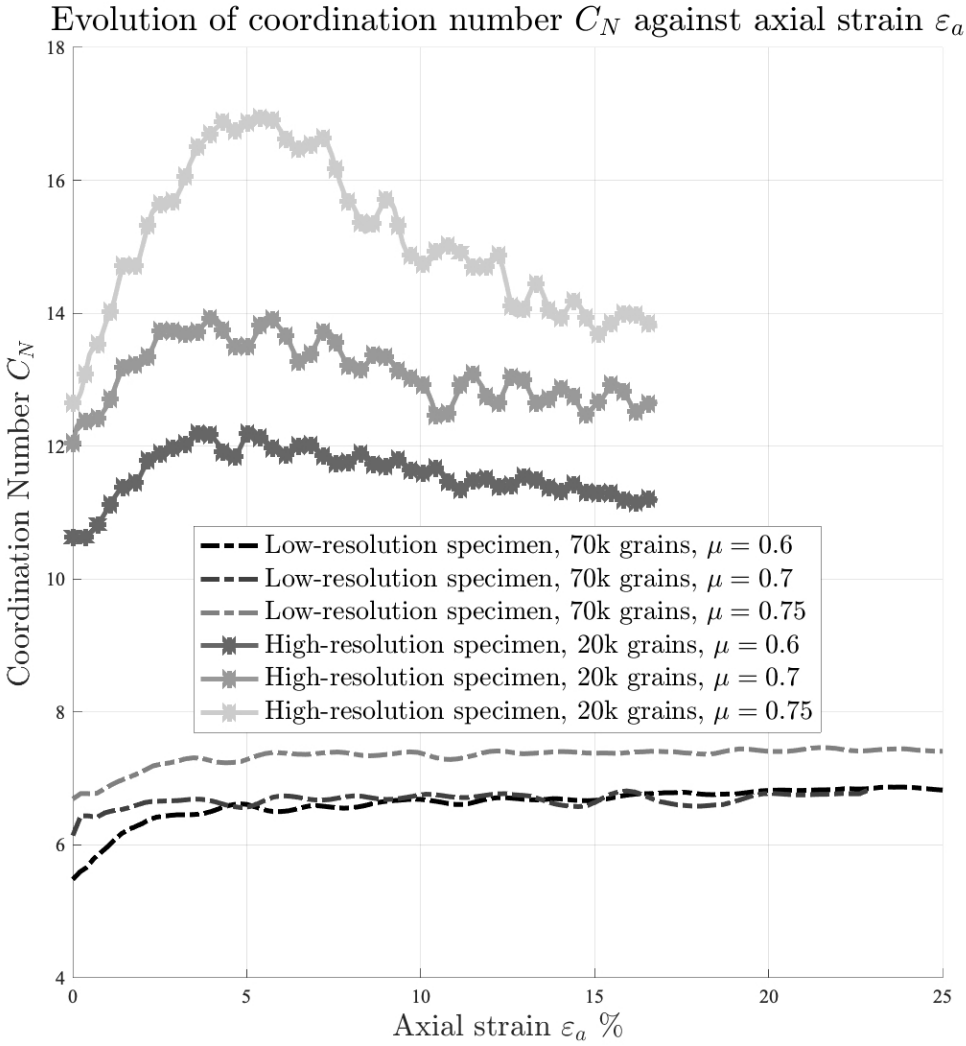

Figure 4.14: Evolution of coordination number during deviatoric loading.

4.6.3 Shear Band Evolution

Compared to conventional servo method using rigid boundary, one outstanding advantage of the flexible membrane is the development of a shear band. The shear band is featured

by large amount of localized shear deformation when granular material is deformed beyond its elastic limit, and it is an indicator the instability of triaxial specimen as the initial local inhomogeneity in the specimen give rise to concentrating deformation in its vicinity (Rudnicki & Rice, 1975; Rice, 1976). One common measurement of the shear band is its thickness in terms of multiples of the mean grain diameter D_{50}, such value is found varying with grain shape, specimen porosity and mean stress (Guo, 2012) and the range of t_s is found from $10D_{50}$ (Kawamoto et al., 2018), $30 \sim 60D_{50}$ (Mollon et al., 2020), $9 \sim 14D_{50}$ (Amirrahmat et al., 2019) and $10 \sim 15D_{50}$ in our numerical results (Figure 4.16) and $8 \sim 12D_{50}$ in our experimental data.

The shear band inclination angle is determined by first locating grains with large rotations or local deviatoric strains greater than two standard deviations over the average values, and then locating the plane with normal direction and height that minimizes the total sum of squared distances between previously identified grains and the plane. This is a least squares problem with outliers, and an orthogonal matching pursuit (OMP) algorithm is used to eliminate those outliers: grains that are far from the shear band but rotate significantly as the result of conforming to the loading platen's motion. In terms of the angle of shear band inclination, the Roscoe's theory predicts it (Roscoe, 1970) as:

$$\theta_R = 45^\circ + \frac{\psi_{\max}}{2} \tag{4.24}$$

where $\psi_{\max}$ is the maximum dilatancy angle in the zone of deformation that achieves at peak stress. Our simulated data computes the friction angle between $52 \sim 54^\circ$, which is smaller than the Roscoe's solution (61°). This may be due to the non-associativity between stress and strain in granular materials.

The shear band formations of the high-resolution specimen with different internal friction coefficient ($\mu = 0.60,\ 0.70,\ 0.75$) are similar. The development of shear band begins with a diffusive pattern prior to the formation of one particular shear band as displayed in Figure 4.15. Following the peak stress, one of the shear bands is activated and forms a major shear band. Micro shear bands are detected during the hardening phase at axial shortening $\varepsilon_a = 5\%$, with a typical form originates from the upper part and then slants downward through the cylindrical specimen. As the simulation progresses into the softening phase at $\varepsilon_a = 10\%$, more micro shear bands are developed parallel or perpendicular to the direction of the final shear band, leading to a thicker, more extending region. With the ongoing compression ($\varepsilon_a = 15\%$), the micro shear bands merge to form a zone of intensive shearing at the center of the specimen whose orientation is preferentially aligned with the final major shear band. Continuing through the critical state, the final major shear band is fully developed and has a thickness of $t_s = 10 \sim 15D_{50}$, as displayed in Figure 4.16.

For the low-resolution specimen, the evolution of strain localization is displayed in Figure 4.17. A single shear band is still identified but the upper platen was tilted by a large amount to create a more pronounced strain localization. Figure 4.18 depicts the contact network distributions and microscale normal force chains for a low-resolution specimen with an axial shortening of between $\varepsilon_a = 0 \sim 30\%$. At the onset of compression, a more randomly

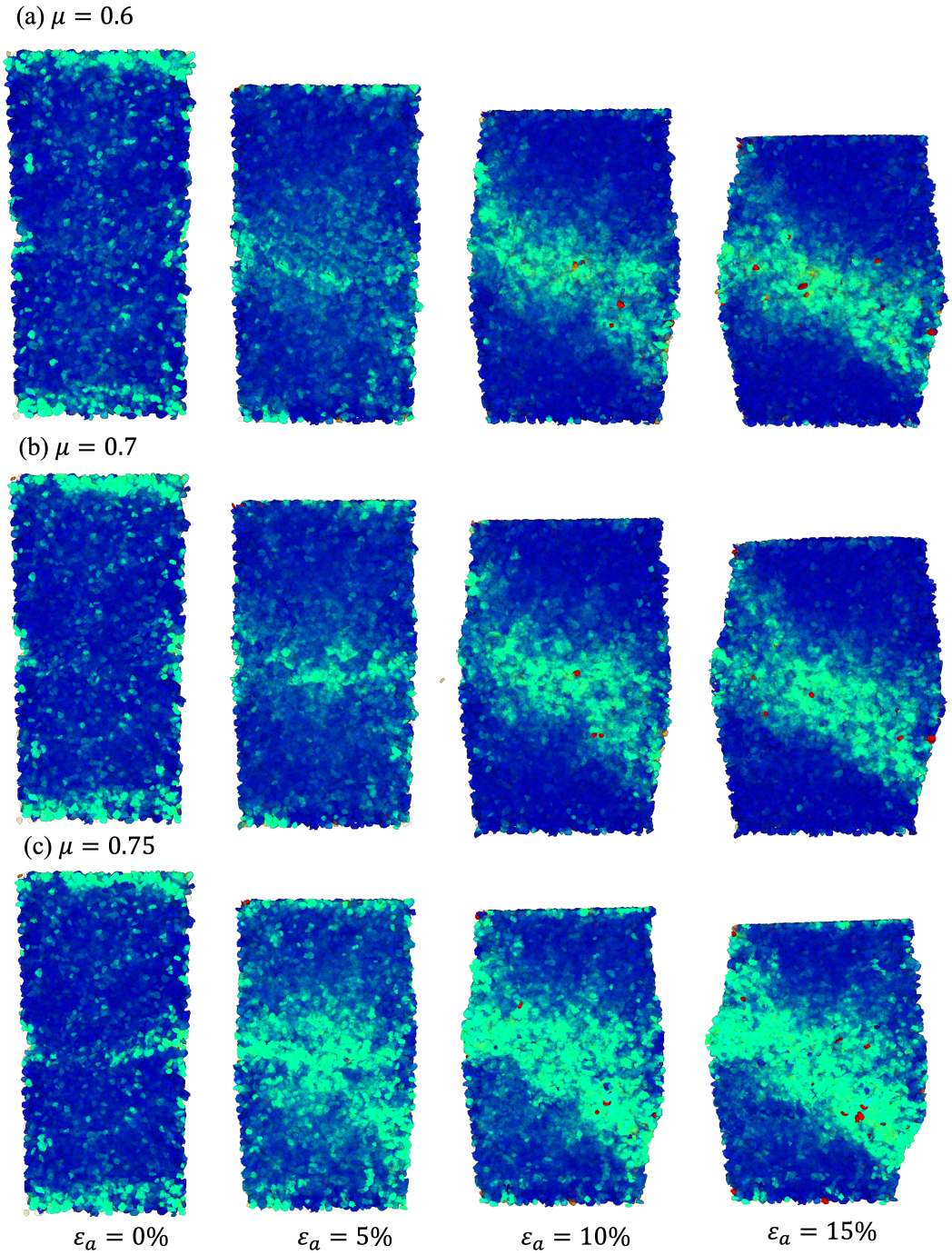

Figure 4.15: Single shear band pattern obtained from high-resolution specimen between $0 \sim 15\%$ axial shortening with internal friction coefficient, (a) $\mu = 0.6$; (b) $\mu = 0.7$; (c) $\mu = 0.75$.

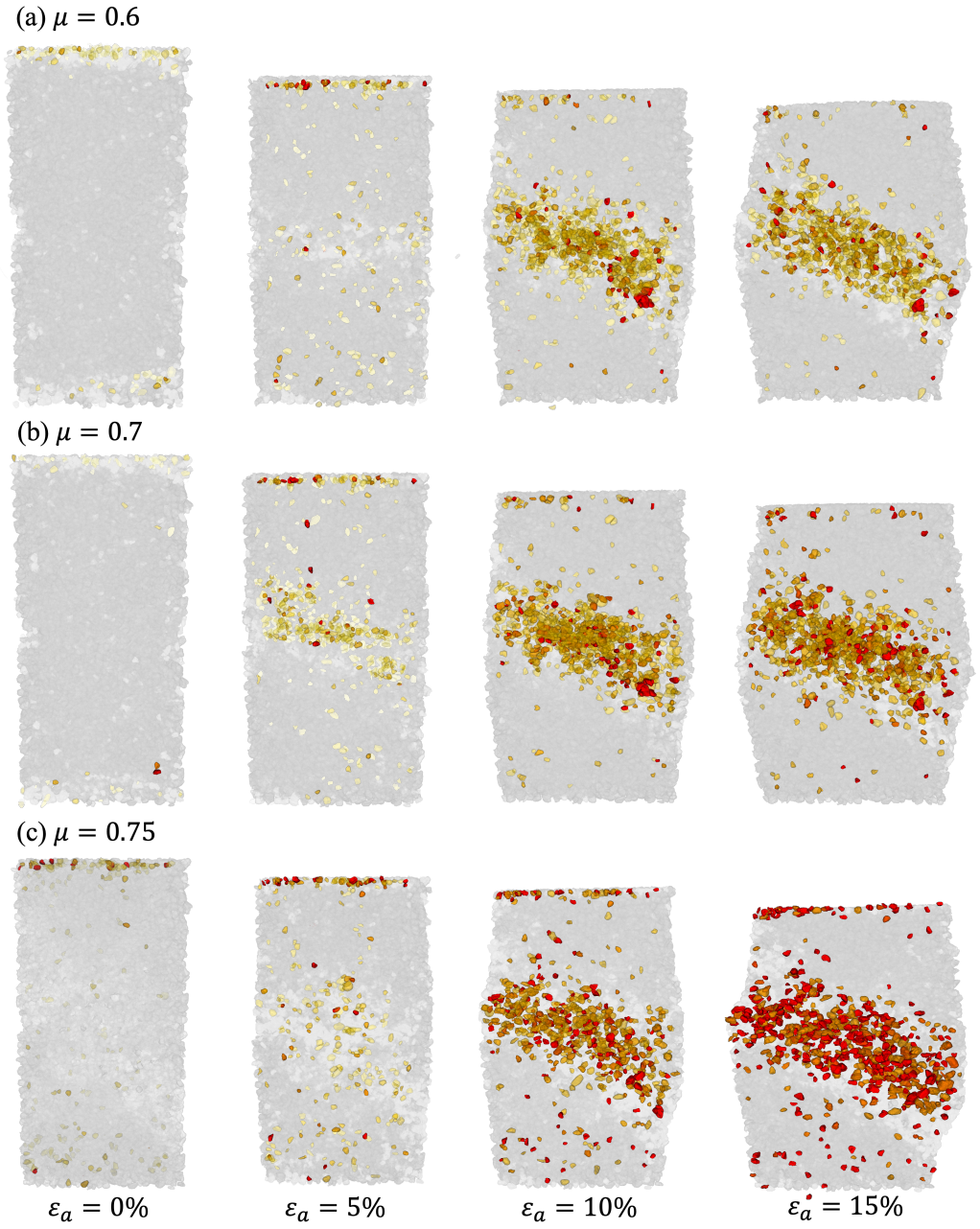

Figure 4.16: Grains with rotation greater than the mean rotation by two standard deviations (2σ) obtained from high-resolution specimen between $0 \sim 15\%$ axial shortening with internal friction coefficient, (a) $\mu = 0.6$; (b) $\mu = 0.7$; (c) $\mu = 0.75$.

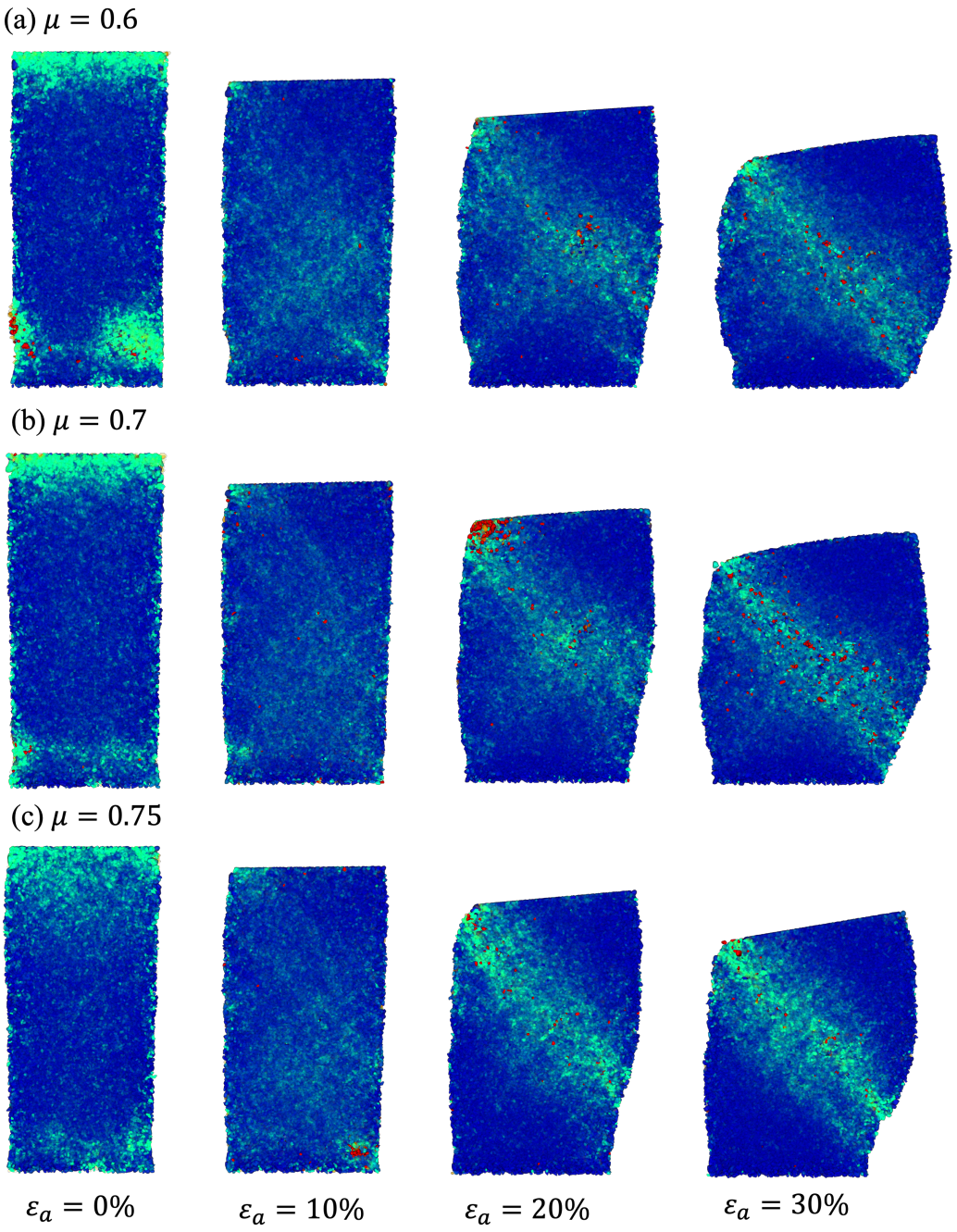

Figure 4.17: Single shear band pattern obtained from low-resolution specimen between 0 $\sim$ 30% axial shortening with internal friction coefficient, (a) $\mu = 0.6$; (b) $\mu = 0.7$; (c) $\mu = 0.75$.

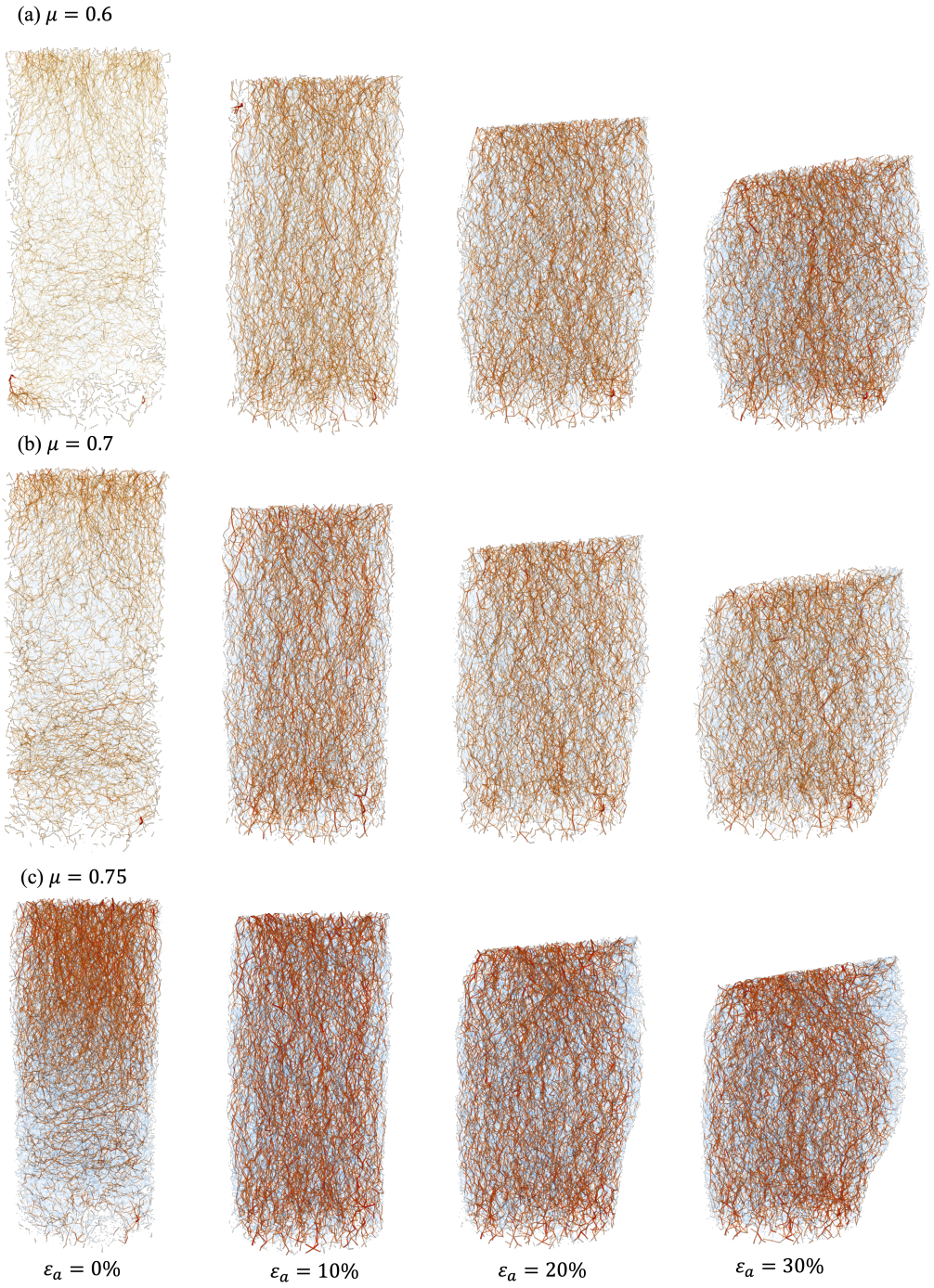

Figure 4.18: Force chain evolution obtained from low-resolution specimen between $0 \sim 30\%$ axial shortening with internal friction coefficient, (a) $\mu = 0.6$; (b) $\mu = 0.7$; (c) $\mu = 0.75$, forces change directions when they passing through the shear band.

oriented homogeneous distribution of contact force networks is observed following isotropic consolidation. In the subsequent deviatoric stages, stronger contact forces concentrate near the specimen's center axis and align vertically to resist the applied axial stress. Numerous strong force chains are identified that can withstand increased loading and contribute to the enhanced macro-scale stress-strain behavior. These chains buckle and restructure in subsequent stages as grains displace and rotate. When the specimen reaches the critical state, the forces in the shear band and out of the shear band appear to lose coaxiality, where force chains seem to change direction as they pass through the shear band. Several stronger force chains remain but are congregated toward the specimen's center, and more lateral contacts are formed and stressed.

Interestingly, the strain localization pattern can change with different parameter settings for the low-resolution specimen. According to published research (Amirrahmat et al., 2019; Mollon et al., 2020; Rorato et al., 2021), shear band simulations are just as prone to variability as actual experiments, and future research should focus on the repeatability of such simulations. The packing density, boundary conditions, and confining pressure all contribute significantly to mechanical interlocking, relative grain translation, and the tendency to develop a shear band. We further investigated different strain localization pattern, Figure 4.19 and Figure 4.20 show the shear pattern and the force chain for low-resolution specimen with the same confining pressure, but set the membrane sphere-sphere normal stiffness and tangential stiffness to be $K_{nbb} = K_{sbb} = 100N/m$. For this case, randomly oriented micro shear bands congregated into a centralized zone at central part of the specimen ($\varepsilon_a = 10\%$). As shearing progressed at $\varepsilon_a = 20\%$, a more complex network of micro shear bands forms in conjugate directions, this leads to a centralized cross-like strain localization pattern that dominates the internal microstructure and forces grains outward in the lateral direction, resulting the surface bulging of specimen. When the stress plateau is reached, the cross-like pattern of localization persists and becomes more defined and stable; however, the thickness of two shear bands varies significantly, with one being easily identifiable and thinner and the other being more diffusive but thicker.

4.6.4 Evolution of the Soil Fabric

We use the anisotropy factor, deviatoric fabric tensor and distribution density of directionality to investigate the evolution of soil fabric when a intensive strain localization pattern forms. In this section, we take high-resolution specimen with $\mu = 0.75$ as an example and we define the representing orientation as the direction of the longest axis of a grain. The grains outside shear region for various shear stage are calculated with the shear band location at 16% axial shortening. The general pattern is that the soil fabric in both windows remains unchanged prior to strain localization and undergoes distinct evolutions afterwards. Among numerous descriptors used to study the evolution of fabric, anisotropy is a scalar that describes the spread of the orientation. For an isotropically distributed discrete system, the anisotropy factor approaches zero. In our study, the anisotropy factor is found to be equal and non-zero ($a = 0.41 \sim 0.42$) both inside and outside the shear band at the onset of simu-

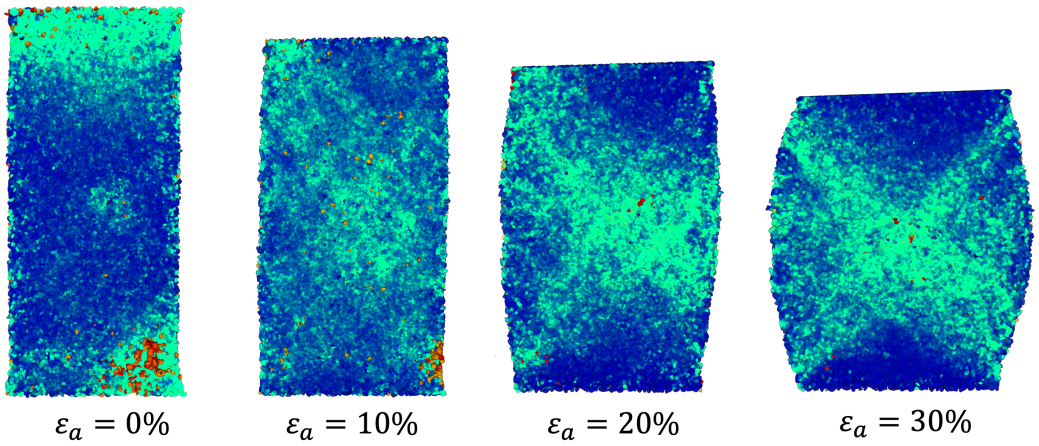

Figure 4.19: Cross-like localization pattern obtained from low-resolution specimen between $0 \sim 30\%$ axial shortening, with $K_{nbb} = K_{sbb} = 100N/m$.

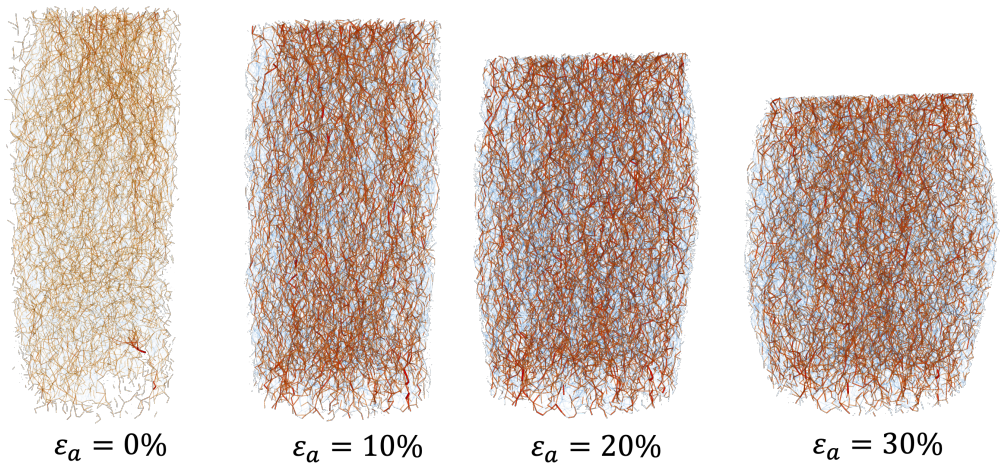

Figure 4.20: Evolution of force chain for low-resolution specimen between $0 \sim 30\%$ axial shortening, with $K_{nbb} = K_{sbb} = 100N/m$.

lation, implying that the initial fabric is anisotropic due to the fact that the major principal orientations of grains are more likely to align perpendicular to the direction of deviatoric stress for a more stable state. By contrast, Wiebicke et al. (2020) reported that a rounded sample prepared by air pluviation with grains falling vertically exhibits little anisotropy in the initial fabric and a small initial inclination angle.

The evolution of the anisotropy factor is monotonically increasing for both windows inside and outside the shear band, whereas the changes in inclinations are much faster inside the shear band whereby grains rearrange in the direction in which the shear band is forming under large shearing. Another descriptor, the deviatoric tensor, which is the deviatoric component of the fabric tensor, is plotted in three dimensions in Figure 4.21 and Figure 4.22. Before the onset of strain localization, the soil fabric tensors are more spherical as also indicated from the anisotropy factor. While the anisotropy factor is nearly identical, the tensor shapes are slightly different. This demonstrates the need to characterize soil fabric with more than just a scalar anisotropy. With ongoing shear within the shear band, the deviatoric tensor evolves into a peanut shape. Outside the shear band, the deviatoric fabric exhibits a similar trend but is much less pronounced. A spherical histogram can be used to quantify and visualize the directionality of grain orientations. Between 0% and 16% axial shortening, Figure 4.23 and Figure 4.24 illustrate the evolution of the distribution of long axes of grains inside and outside the specimen's shear band. Within the shear band, the grains undergo a significantly greater rotations as loading progresses than those outside the shear band. As is the case with the anisotropy scalar and fabric tensor, the directionality does not change significantly prior to the peak ($\varepsilon_a = 5\%$). Following that, grains within the shear band unlock, rotate, and align in a well-defined direction in favor to the formation of shear band. While grains outside the shear band exhibit a similar trend, it is less pronounced and more localized in nature, with only comparably much small and random kinematic outside the shear band. This might be due to the fact the dilated zone is much larger than the area where shear and grain rotation actually occurs, suggesting that grains dilate and rotate relative to one another in a band wider than the shear region (Mollon et al., 2020). Additionally, grains outside the shear band will act as a single entity that will displace and rotate in response to the shear deformation, this would also slightly alter the grain's orientation.

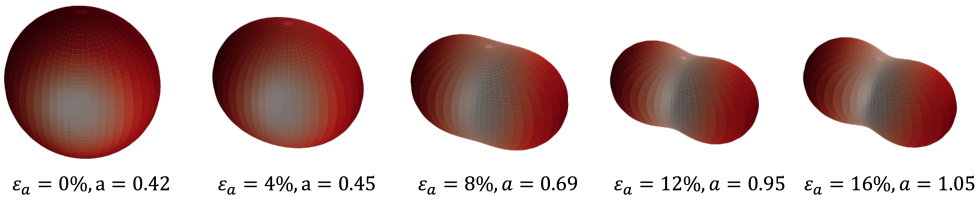

Figure 4.21: Surface plots of the distribution density of deviatoric fabric tensor and the anisotropy factor of grain inside the shear band orientation.

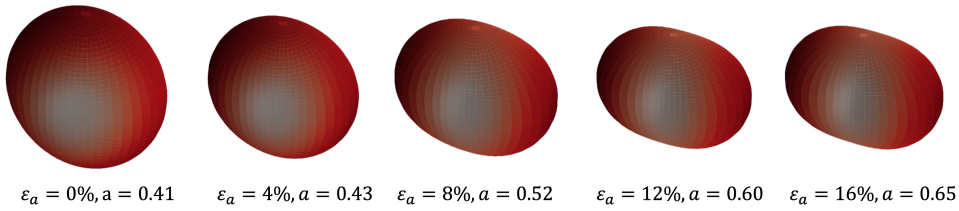

Figure 4.22: Surface plots of the distribution density of deviatoric fabric tensor and the anisotropy factor of grain outside the shear band orientation.

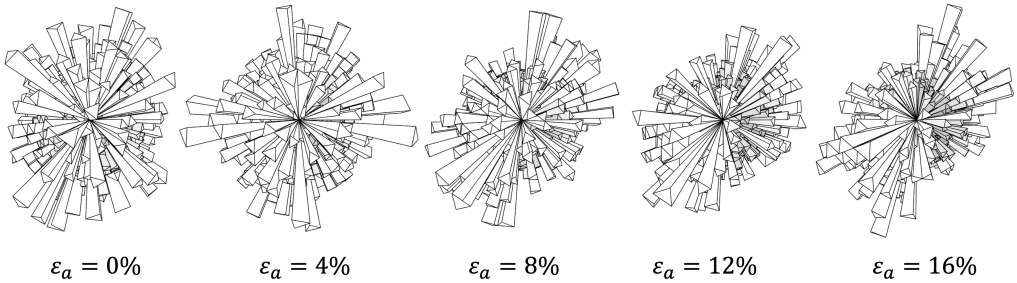

Figure 4.23: Spherical histogram of grain inside the shear band orientation.

4.6.5 Strain-Controlled Cyclic Loading Tests

we also explored LS-DEM simulations for compression tests on granular assembly subjected to a sequence of loading and unloading cycles, with strain-controlled stress relaxation (Figure 4.25). One objective of this simulation was to replicate the stress state of the specimen in the laboratory, which was stopped and imaged at various stations during loading. According to Kawamoto et al. (2018), no significant granular rearrangement occurs, confirming that interrupting axial loading for imaging has a negligible effect on experimental results. Because the amount of reversible strain is small and the loading stress is sufficiently high (to achieve a mobilized friction angle greater than 45°, the axial loading should be 6 times greater than the confining pressure), the mechanical behavior of the packing tends to be elastic. When loading stresses are low, as they are during the initial stage of compression, the assembly's internal structural evolution is likely to be dominated by irreversible porosity reduction caused by pore collapse and grain re-arrangement, and as a result, the assembly is unlikely to fully restore its volume. When loading stresses become sufficiently high but not yet close to the threshold for grain breakage, the room for residual porosity to diminish further is significantly reduced, resulting a rather elastic response.

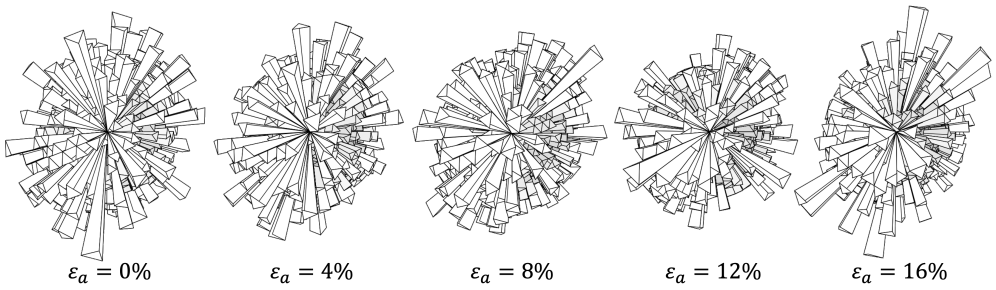

Figure 4.24: Spherical histogram of grain outside the shear band orientation.

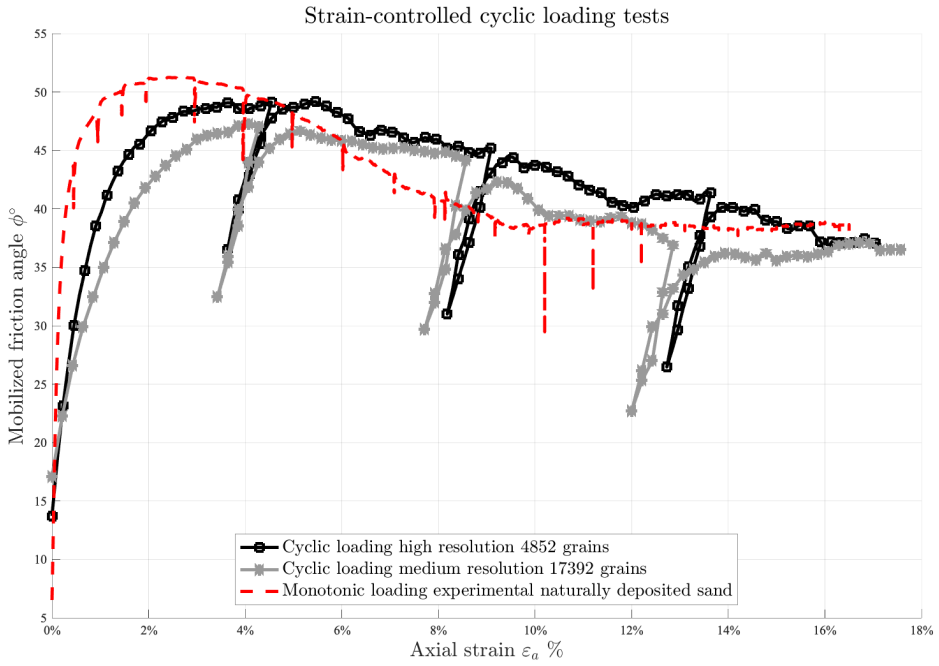

Figure 4.25: Mobilized friction angle of two numerical specimens under strain-controlled loading and unloading cycles.

4.6.6 Computational Effort

As previously demonstrate, the computational cost is primarily determined by the number of discretized nodes used to represent grain shape, and the increasingly accurate capture of grain shape comes with substantially less grain being reconstructed from a fixed-size window of XRCT scans. Fortunately, our parallelized code is sufficiently efficient in that it can solve problems of comparable size in a comparable amount of time as the originally proposed

code by Kawamoto et al. (2016), while consuming less than 5% of computing resources. In current study, our code simulated an assembly of $\sim 70,000$ low-resolution grains with 250 nodes per grain in the same amount of time as a high-resolution assembly of $\sim 20,000$ grains with 750 nodes per grain. This demonstrates that the actual factor controlling the execution speed off code is proportional to the number of nodes on the grain's surface. Thus, we found simulating an even higher resolution assembly containing only $1,600$ grains with more than $2,500$ discretized nodes on each is numerically an almost intractable task.

4.7 Conclusion

LS-DEM simulations of naturally deposited uncemented sands subjected triaxial compression have been presented in this study. A linear elastic model with Coulomb friction yield criterion was calibrated, verified, and used to simulate the frictional response of an assembly of LS avatars, which are one-to-one direct mappings from XRCT scans and capable of preserving as much as possible grain morphology and surface roughness. A significant conclusion reached in this study is that the primary source of frictional resistance is mechanical interlocking between irregularly shaped grains; thus, even the simplest contact model is capable of reproducing both micro- and macromechanical responses consistent with experimental data, provided that the microstructure of the grain is captured with sufficient fidelity using high quality scans. Flexible membrane simulations with a rotatable loading platen were found to better predict not only stress-strain and volumetric response, but also the onset and growth of strain localization, allowing them to accurately match experimentally observed relationships between deviatoric stress and mobilized friction angle with axial shortening for uncemented sample.

LS-DEM quantifies the kinematic behavior of real-world grain shapes and enables the study of intricate strain localizations down to the thickness and delineation level, which were previously rarely studied in conventional DEM modeling trivial geometries. Monitoring grain rotations during virtual tests aids in identifying high-shearing zones and provides insight into the failure mode of granular media, e.g., barreling failure or shear banding. Low- and high-resolution specimens with the same initial void ratio and confining stress exhibit distinct strain localization patterns, with complex hourglass-like patterns for the low-resolution reconstruction and a well-defined congregated shear localization for the high-resolution reconstruction. In comparison to macroscopic frictional responses, grain-scale responses in shear band simulations are subject to considerable uncertainty.

The coordination number, anisotropy factor, directionality, and a second-order deviatoric fabric tensor are used to analyze the evolution of the soil fabric. The high-resolution specimen has a higher initial coordination number and is therefore more difficult to rotate, it also results in increased frictional resistance and significant volumetric dilation. Contacts form and break dynamically in low-resolution specimens and balance each other at high shear after reaching the critical state. Following the onset of strain localization, the soil fabric inside and outside the shear band evolves in distinct ways, with anisotropy increasing and orientations aligning

with the major shear band direction inside the shear band. Both evolutions are predicted by micromechanical analysis of grain kinematics. The micro analysis revealed that the major kinematic changes, particularly grain rotations, occur within the shear band, while much smaller and random kinematic changes occur outside the shear band.

While these findings are significant and necessary for a complete micromechanical description, they must be interpreted cautiously. To begin, contact properties are still determined in a heuristic manner, and they vary according to the type of sand. This necessitates the replication and extension of experimental studies under varying initial conditions in order to validate and complement these findings. Second, the predictive power of LS-DEM should be investigated under a variety of loading conditions, as the triaxial compression test has been almost exclusively used to study grain microstructure in current study. As a trial, our study has been extended to cyclic loading with strain relaxation, with expected results, this piques interest in more sophisticated loading conditions. Finally, and perhaps most importantly, the quality of the images used to generate the LS correspondences is critical to the success of the simulation on both the micro- and macro-mechanical response of granular assembly.

Chapter 5

Impulse-Based LS-DEM for Dynamic Problems

5.1 Introduction

Typical DEM codes in general use in research and in the industry use penalty-based approach to resolve forces between the discrete bodies, which are then converted to forces, accelerations, and finally displacements. This model has varying complexity to generate a contact interaction force based on separation distance between particles in contact (Zohdi, 2017). However, they are computationally demanding, if not prohibitively expensive, for large-scale DEM simulations with complex-shaped grains or particles. While recent research efforts have been made in hardware to accelerate these DEM codes using high-performance computing clusters or graphics processing units (GPUs) (Owen & Feng, 2001; Zheng et al., 2012; Yan & Regueiro, 2018b; Yan & Regueiro, 2018a) such resources are not generally or broadly accessible.

Alternatively, an impulse-based technique based on Newton's impact law is significantly more computationally efficient since it directly computes the velocity increment of the modeled objects. It was originally designed for the computer graphics industry (Mirtich & Canny, 1995; Chang & Colgate, 1997; Bender, 2007) and prioritizes code speed, stability, and physical plausibility over simulation fidelity. Recent work (Tang et al., 2014; Lee & Hashash, 2015; Asai et al., 2021; Li et al., 2021) demonstrates that, in addition to much improved code speed and numerical stability, the impulse-based technique also produces results with comparable accuracy to the corresponding penalty-based DEM simulations. This makes the impulse-based approach intriguing since it is capable of addressing very large dynamic problems.

Herein we summarize the framework of impulse-based rigid body dynamics, which is fundamentally a process of kinetic energy conversion to elastic energy. Rather than providing a broad introduction to the subject, we focus on the numerical aspects of the impulse-based approaches and the possibility of parallelizing impulse-based LS-DEM using the domain

decomposition approach. We show that in comparison to parallel implementation for penalty-based LS-DEM, impulse-based DEM saves considerable effort in data communication but requires additional handling of multibody assemblies in direct contact. We then introduce a modified Energy Tracking Method (ETM, Tang et al., 2014) in order to resolve collisions within the domain with a wide variety of grain shapes and sizes and to add deformable structures into rigid body dynamics. Finally, we illustrate the performance and applicability of the impulse-based LS-DEM for rock fall and rock avalanche simulations.

5.2 Formulation of Impulse-Based Rigid Body Dynamics

A negative relative velocity between two objects at the time of a collision suggests that they are about to penetrate. To avoid penetration between rigid bodies, the negative relative velocity is increased till it reaches zero by applying a sequence of impulses in the normal direction of contact. The compression phase converts kinetic energy to elastic energy, which is then stored at the contact points. Following that, previously held elastic energy is released to restore relative normal velocity, ejecting two bodies from one another and concluding the collision process. As it is almost impossible to predict the microstructure of materials during collisions in practice, we adopt Stronge's hypothesis (Stronge, 2018): the amount of energy absorbed and released may vary according to the restitution coefficient. For example, for a perfectly elastic collision with no energy loss. Stronge's hypothesis states that energy is dissipated as follows:

$$W_{release} = \epsilon^2 W_{absorbed} \tag{5.1}$$

Where $W_{release}$ and $W_{absorbed}$ denote the energy released and absorbed during release and compression phases, respectively. $\epsilon (0 \leq \epsilon \leq 1)$ denotes the coefficient of restitution that determines the elasticity of the contact. Without loss of generality, we formalize a single collision process by considering a body b_A collides with another b_B due to the i-th impulse $\mathbf{P}^i$, which effectively changes their linear velocities ($\mathbf{V}_A$ and $\mathbf{V}_B$) and angular velocities ($\boldsymbol{\omega}_A$ and $\boldsymbol{\omega}_B$), as shown in Figure 5.1 via:

$$\mathbf{P}^i_A = M_A \Delta \mathbf{V}_A \tag{5.2}$$

$$\mathbf{P}^i_B = M_B \Delta \mathbf{V}_B \tag{5.3}$$

$$\mathbf{r}^i_A \times \mathbf{P}^i_A = \mathbf{I}_A \Delta \boldsymbol{\omega}_A \tag{5.4}$$

$$\mathbf{r}^i_B \times \mathbf{P}^i_B = \mathbf{I}_B \Delta \boldsymbol{\omega}_B \tag{5.5}$$

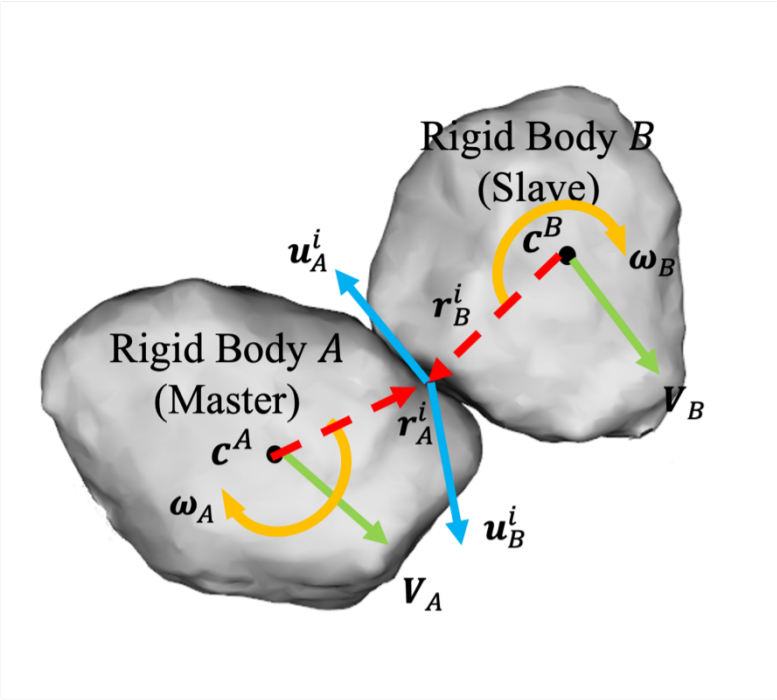

Figure 5.1: Collision between bodies b_A and b_B: $\mathbf{V}_A$ and $\mathbf{V}_B$ are linear velocities of the center of mass, $\boldsymbol{\omega}_A$ and $\boldsymbol{\omega}_B$ are angular velocities, $\mathbf{r}_A^i$ and $\mathbf{r}_B^i$ are the vectors from the center of mass of bodies to the contact point where the i-th impulse $\mathbf{P}^i$ applied, $\mathbf{u}_A^i$ and $\mathbf{u}_B^i$ are the velocities at the contact points, and $\mathbf{n}^i$ is the contact normal.

$$\mathbf{P}_A^i = -\mathbf{P}_B^i \tag{5.6}$$

Where M_A and M_B are the masses, I_A and I_B are the inertial tensors, $\mathbf{r}_A^i$ and $\mathbf{r}_B^i$ are the relative positions from the center of gravity of b_A or b_B to the contact point at which the i-th impulse was applied. The velocities $\mathbf{u}_A^i$ and $\mathbf{u}_B^i$ of contact points on b_A and b_B in global frame are:

$$\mathbf{u}_A^i = \mathbf{V}_A + \boldsymbol{\omega}_A \times \mathbf{r}_A^i \tag{5.7}$$

Therefore, the velocity changes at a point of contact between body b_A and body b_B due to the i-th impulse respectively is given below. Here, we suppose that the collision occurs and resolves quickly enough that the contact point i remains stationary.

$$\Delta\mathbf{u}_A^i = \Delta\mathbf{V}_A + \Delta\boldsymbol{\omega}_A \times \mathbf{r}_A^i = \frac{1}{M_A}\mathbf{P}_A^i + \mathbf{I}_A^{-1}\mathbf{r}_A^i \times \mathbf{P}_A^i \times \mathbf{r}_A^i = \left(\frac{1}{M_A} - \tilde{\mathbf{r}}_A^i \mathbf{I}_A^{-1} \tilde{\mathbf{r}}_A^i\right)\mathbf{P}_A^i \tag{5.8}$$

$$\Delta\mathbf{u}_B^i = \Delta\mathbf{V}_B + \Delta\boldsymbol{\omega}_B \times \mathbf{r}_B^i = \frac{1}{M_B}\mathbf{P}_B^i + \mathbf{I}_B^{-1}\mathbf{r}_B^i \times \mathbf{P}_B^i \times \mathbf{r}_B^i = \left(\frac{1}{M_B} - \tilde{\mathbf{r}}_B^i \mathbf{I}_B^{-1} \tilde{\mathbf{r}}_B^i\right)\mathbf{P}_B^i \tag{5.9}$$

The second equality converts cross product between vectors into equivalent matrix-vector multiplication. In the vector space $\mathcal{R}^3$, the cross produce ($\times$) is an operator taking two vectors to a third vector:

$$\mathbf{a} \times \mathbf{b} = \begin{pmatrix} a_y b_z - a_z b_y \\ a_z b_x - a_x b_z \\ a_x b_y - a_y b_x \end{pmatrix} = \tilde{\mathbf{a}}\mathbf{b} \tag{5.10}$$

Where $\tilde{\mathbf{a}}$ is a 3×3 skew-symmetric matrix. In the collision formula above:

$$\tilde{\mathbf{r}} = \begin{pmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{pmatrix} \tag{5.11}$$

Above formulas imply that if we know the change of relative velocity at the contact points, the resulted impulse can be computed via:

$$\Delta\left(\mathbf{u}_A^i - \mathbf{u}_B^i\right) = \Delta\mathbf{u}_A^i - \Delta\mathbf{u}_B^i = \left[\left(\frac{1}{M_A} + \frac{1}{M_B}\right)\mathbf{I} - \left(\tilde{\mathbf{r}}_A^i \mathbf{I}_A^{-1} \tilde{\mathbf{r}}_A^i + \tilde{\mathbf{r}}_B^i \mathbf{I}_B^{-1} \tilde{\mathbf{r}}_B^i\right)\right] \mathbf{P}_A^i \tag{5.12}$$

$$\mathbf{K} = \left(\frac{1}{M_A} + \frac{1}{M_B}\right)\mathbf{I} - \left(\tilde{\mathbf{r}}_A^i \mathbf{I}_A^{-1} \tilde{\mathbf{r}}_A^i + \tilde{\mathbf{r}}_B^i \mathbf{I}_B^{-1} \tilde{\mathbf{r}}_B^i\right) \tag{5.13}$$

Where $\mathbf{P}_A^i$ is the i-th impulse exerted on the body b_A, and $\mathbf{K}$ is termed as the collision matrix and it is non-singular, symmetric, positive definite and defined in global frame for a given collision (Mirtich, 1996). Collision contact occurs within a very short time interval for a rigid body, implying that displacement and change in the contact area are insignificant. This indicates that the collision matrix $\mathbf{K}$ remains constant throughout the collision, and that once the change in relative velocity is known, the collision matrix is used to compute the impulses:

$$\mathbf{P}^i = \mathbf{K}^{-1} \Delta\mathbf{u}^i \tag{5.14}$$

Where $\mathbf{P}^i = \mathbf{P}_A^i = -\mathbf{P}_B^i$ is the i-th impulse, $\mathbf{u}^i = \mathbf{u}_A^i - \mathbf{u}_B^i$ is the relative velocities at the contact points where the i-th impulse is applied, and $\Delta\mathbf{u}^i$ is the change of relative velocities. This formula remains true if the friction forces are ignored or if the friction forces are constant. As a result, if there is no slip between the bodies, only the normal component of the change in relative velocity $\Delta\mathbf{u}_n^i$ is non-zero, because it is the relative normal velocity at the contact locations that drives the bodies to collide. Thus, the preceding formula is reduced to:

$$\mathbf{n}^i \cdot \mathbf{K}\mathbf{P}^i = \mathbf{n}^i \cdot \Delta\mathbf{u}^i = \Delta\mathbf{u}_n^i \tag{5.15}$$

$$\left|\mathbf{P}_n^i\right| = \mathbf{n}^i \cdot \mathbf{P}^i \mathbf{n}^i \tag{5.16}$$

$$\mathbf{P}_t^i = \mathbf{P}^i - \mathbf{P}_n^i \tag{5.17}$$

If sliding occurs, apart from ensuring that relative normal velocity changes are consistent with the prescribed value, the impulse must be recalculated to meet Coulomb's friction requirement, which limits the magnitude of tangential impulse:

$$\left|\mathbf{P}_t^i\right| = \mu \left|\mathbf{P}_n^i\right| \tag{5.18}$$

$$\mathbf{P}^i = \left|\mathbf{P}_n^i\right| \mathbf{n}^i + \mu \left|\mathbf{P}_n^i\right| \mathbf{t}^i \tag{5.19}$$

$$\left|\mathbf{P}_n^i\right| = \frac{\Delta \mathbf{u}_n^i}{\mathbf{n}^i \cdot (\mathbf{K}\mathbf{n}^i + \mu \mathbf{t}^i)} \tag{5.20}$$

Where $\mathbf{n}^i$, $\mathbf{t}^i$ are the contact normal and tangential direction at the i-th collision, and $\mathbf{P}_n^i$ and $\mathbf{P}_t^i$ are the normal and tangential components of applied impulse. This concludes the compression phase where the corresponding impulse is generated based on the change of relative velocity and the kinetic energy is stored in the form of elastic energy at the contact.

The next step is to dissipate the elastic energy and invert the sign of the relative normal velocity to separate the bodies. This is the inverse of the compression phase: a predefined amount of elastic energy is dissipated to increase the relative normal velocity via a series of impulses. Mirtich (1996) proved that if the relative contact velocity $\mathbf{u}^i$ proceeds from $\mathbf{u}_0^i$ to $\mathbf{u}_f^i$ during a collision and over some arbitrary path, the total work done by the collision impulse $\mathbf{P}^i$ is independent of the path taken and is given below, which enables the change in relative velocities to be calculated from the released elastic energy.

$$\Delta W^i = \frac{1}{2}\left(\mathbf{u}_f^i + \mathbf{u}_0^i\right)^T \mathbf{K}^{-1}\left(\mathbf{u}_f^i - \mathbf{u}_0^i\right) = \frac{1}{2}\left(\mathbf{u}_f^i + \mathbf{u}_0^i\right)^T \mathbf{P}^i \tag{5.21}$$

Given that rigid body motion is better defined in terms of its principal or local frame, it is more convenient to address collisions in the local coordinate system via the rotation matrix $\mathbf{R}$, which defines the transformation from body to global coordinates. In implementation, rotations are handled numerically via a singularity-free quaternion technique.

$$\Delta W^i = \frac{1}{2}\left(\mathbf{u}_f^i + \mathbf{u}_0^i\right)^T \mathbf{P}^i = \frac{1}{2}\left(\mathbf{R}\tilde{\mathbf{u}}_f^i + \mathbf{R}\tilde{\mathbf{u}}_0^i\right)^T \mathbf{R}\tilde{\mathbf{P}}^i = \frac{1}{2}\left(\tilde{\mathbf{u}}_f^i + \tilde{\mathbf{u}}_0^i\right)^T \tilde{\mathbf{P}}^i \tag{5.22}$$

Where $\tilde{\mathbf{u}}_0$, $\tilde{\mathbf{u}}_f$ and $\tilde{\mathbf{P}}^i$ denote initial relative velocity, final relative velocity, and applied impulse, all with respect to the body frame. During a collision, the work done by the impulse can be decomposed into the work done by the impulse in the normal direction ΔW_n^i and the work done by the impulse in the tangential direction ΔW_t^i.

$$\Delta W^i = \Delta W_n^i + \Delta W_t^i \tag{5.23}$$

$$\Delta W_n^i = \frac{1}{2}\left(2\tilde{u}_n^i + \Delta\tilde{u}_n^i\right)\tilde{P}_n^i \tag{5.24}$$

$$\Delta W_t^i = \frac{1}{2}\left(2\tilde{u}_t^i + \Delta\tilde{u}_t^i\right)\tilde{P}_t^i \tag{5.25}$$

Where $\tilde{u}_n^i$, $\tilde{u}_t^i$ and $\tilde{P}_n^i$, $\tilde{P}_t^i$ are the normal and tangential component of relative velocities and impulses at contact points defined in the body frame. It is more usual to depict a plane perpendicular to the contact normal using two orthonormal vectors, which is also compatible with the rotation operation transferring three orthonormal vectors into another three. We decided not to differentiate two tangential directions here since we were only concerned with the normal components: tangential velocities would remain unchanged because they did not break the impenetrability requirements between rigid entities. Due to the fact that work performed in different directions is independent, the change in relative normal velocity has no effect on the amount of work performed in the normal direction by the impulse, which later inverts the relative normal velocity by releasing the energy absorbed during the compression phase.

The change of relative velocity $\Delta\mathbf{u}^i$ and corresponding work done ΔW^i by the impulse implies that there should be a way to compute $\Delta\mathbf{u}^i$ (ΔW^i) if the collision work ΔW^i ($\Delta\mathbf{u}^i$) is known. Indeed, changing simply the relative normal velocity results in the generation of tangential impulses and thus the associated work. However, because the work performed in normal directions is decoupled, the change in relative normal velocity $\Delta\mathbf{u}_n^i$ can be retrieved provided that we get access the normal impulse $\tilde{P}_n^i$ and the absorbed/released work caused by $\tilde{P}_n^i$. Additionally, the tangential restitution coefficient is set to zero, which requires calibration but is plausible in many impulse-based simulators. As a result, the collision work ΔW^i and ΔW_n^i are used interchangeably, and we focus on the conversion of kinetic to elastic energy in the contact normal direction. For instance, during the compression phase, the relative normal velocity decreases and eventually converts to the elastic energy:

$$\Delta W_n^i = \frac{1}{2}\left(2\tilde{u}_n^i + \Delta\tilde{u}_n^i\right)\tilde{P}_n^i \tag{5.26}$$

Where $\tilde{P}_n^i$ is the normal component of applied impulse. In the separation phase, the absorbed elastic energy reduces and changes the sign of the relative normal velocity via an impulse along the contact normal:

$$\mathbf{K}\mathbf{P}^i = \Delta\mathbf{u}^i \tag{5.27}$$

$$\mathbf{n}^i \cdot \mathbf{K}\left(\tilde{P}_n^i\mathbf{n}^i\right) = \mathbf{n}^i\Delta\mathbf{u}^i = \Delta\tilde{u}_n^i \tag{5.28}$$

$$\tilde{P}_n^i = \frac{\Delta\tilde{u}_n^i}{\mathbf{n}^i \cdot (\mathbf{K}\mathbf{n}^i)} \tag{5.29}$$

$$\Delta\tilde{u}_n^i = -\tilde{u}_n^i + \sqrt{(\tilde{u}_n^i)^2 + 2\Delta W_n^i \mathbf{n}^i \cdot (\mathbf{K}\mathbf{n}^i)} \tag{5.30}$$

5.3 Impulse-Based LS-DEM

5.3.1 Contact Model

Just like in the penalty-based method, contact in impulse-based LS-DEM is handled through node-to-surface contact algorithm and the contact normal direction is also interpolated from the underlying signed distance function grid of a grain. The difference is that contact force calculation is skipped in the impulse-based formulation; however, it could be retrieved from the applied impulse with high-precision. Original LS-DEM (Kawamoto et al., 2016) code considers a history-dependent Coulomb friction model that requires grains to be associated with their contact histories. The inter-grain tangential contact is significantly easier to manage in an impulse-based DEM because it does not track friction evolution and updates velocities directly. During the force resolution phase, the impulse-based DEM iterates through the surface nodes between colliding bodies, identifying all contact points with negative relative velocities, and calculating repulsive impulses to satisfy impenetrability constraints.

The friction type of contact is examined to ensure that the magnitude of the tangential component of the collision impulse follows Coulomb's friction law. Due to the fact that LS-DEM aims to simulate arbitrarily complex grain shapes, it is unavoidable that multiple contact points collide when two reconstructed avatars are on a collision course. To handle many collisions and various contacts, the simultaneous impulse methods (SMM, Barzel & Barr, 1988; Baraff, 1989) ensemble all collisions into a system of linear equations, enforce non-penetration restrictions in rigid body dynamics simulations as a linear complementary problem (LCP) and evaluate outcomes in a single step, SMM fails to capture the propagation of contact forces during a collision. The Sequential Impulse Method (SQM) introduced by Guendelman et al. (2003) resolves each collision within a single time interval and treats the contacts as if they occurred sequentially. This method inverts the relative velocity directly using Newton's law and prioritizes contact points based on the depth of inter-penetration; it is iterative and continues until all contact points have positive relative velocities. The primary disadvantage of SQM is that edge-surface or surface-surface contacts do not end up with similar repulsive velocities and the obtained results dependent on the sequence in which collisions are addressed. To solve these two issues, Tang et al. (2014) and Li et al. (2021) address this issue by gradually changing relative normal velocity and elastic energy in such a way that all contact points with similar normal directions can adjust velocities to similar magnitudes. The compression phase involves incrementally increasing the relative normal velocities at contact points until all relative normal velocities are non-negative. During the separation phase, the absorbed energy is gradually released until no energy remains at the contact points, again in an iterative fashion.

5.3.2 Discrete Equations of Rigid Body Motion in Impulse-Based LS-DEM

Original implementation of LS-DEM (Kawamoto et al., 2016)) solves Newton's and Euler's governing equations of motion for translational and rotational components of motion, respectively. For the translational component of motion, the positions, forces, and velocities are known at the ends of each timestep so that grain motion can be explicitly updated via the governing translational equation given by Newton's law with a centered finite-difference integration scheme. For the rotational components of motion, the time derivatives of the angular accelerations in the principal frame are given by Euler's equations of motion, which is nonlinear and implicit, hence prone to numerical instability.

Again, it is much simpler to change the velocities of two rigid bodies with impulse-based method. For example, if the contact master body b_A experiences a collision contact with b_B under the corresponding impulse $\mathbf{P}^i$, in accordance with Newton's law, the changes in velocity for each rigid body are given by:

$$\Delta\dot{\mathbf{x}}_{A,B} = M^{-1}_{A,B}\mathbf{P}^i_{A,B} \tag{5.31}$$

$$\Delta\dot{\boldsymbol{\theta}}_{A,B} = \mathbf{I}^{-1}_{A,B}\left(\mathbf{r}^i_{A,B} \times \mathbf{P}^i_{A,B}\right) \tag{5.32}$$

The velocities and positions for the rigid bodies are subsequently updated via symplectic Euler scheme:

$$\dot{\mathbf{x}}_{A,B} = \dot{\mathbf{x}}_{A,B} + \Delta\dot{\mathbf{x}}_{A,B} \tag{5.33}$$

$$\dot{\boldsymbol{\theta}}_{A,B} = \dot{\boldsymbol{\theta}}_{A,B} + \Delta\dot{\boldsymbol{\theta}}_{A,B} \tag{5.34}$$

$$\mathbf{x}_{A,B} = \mathbf{x}_{A,B} + \dot{\mathbf{x}}_{A,B}\Delta t \tag{5.35}$$

$$\boldsymbol{\theta}_{A,B} = \boldsymbol{\theta}_{A,B} + \dot{\boldsymbol{\theta}}_{A,B}\Delta t \tag{5.36}$$

5.3.3 Collision Resolution Algorithm

The ETM algorithm is demonstrated with three vertically stacked sphere as shown in Figure 5.2. It treats concurrent collisions as a series of single point collisions and included an additional iteration within a compression and separation phase to make the results insensitive to the order of the impulses. When dealing with numerous contact collisions, ETM gradually delivers impulses to adjust the relative normal velocities of several collisions, yielding low angular velocity errors. Later, Li et al. (2021) adjusted the time step within the sub-cycle loop adaptively to obtain more stable and energy-conservative rigid body dynamic simulations. In impulse-based DEM, if any pair of rigid bodies is connected to a group of LS avatars

via a chain of contacting avatars, collisions at all contact points are resolved simultaneously, as the impulse can propagate within the group. The algorithm begins by iterating through contact points created by collisions and prioritizing those with a negative relative normal velocity. During the compression phase, the priority queue is sorted by minimum relative normal velocity; after the algorithm enters the separation phase, it is sorted by elastic energy. To expedite the process, a group of adjacent contact points is glued together and treated as a single contact. Kinetic and elastic energy are converted to one another during the compression and separation phases. A complete procedure is presented in Algorithm 3.

Algorithm 3 Collision Resolution Algorithm

Input: A list of contacts within a contact island including contact IDs of colliding bodies A and B; branch vectors from centers to contact point $\boldsymbol{r}$; contact normal direction $\tilde{\boldsymbol{n}}$; relative velocity $\tilde{\boldsymbol{u}}$; elastic energy W and body mass M.

Define parameter α to control velocity increment in sub-iteration, Coulomb's friction coefficient μ, similarity β and Stronge's coefficient ϵ.
for all contact points **do**
    Compute inertial matrix in global frame $\boldsymbol{I} = R\boldsymbol{I_0}R^T$, cross-product matrix of colliding bodies $\tilde{\boldsymbol{r}}$, relative normal velocity u_n, collision matrix $\boldsymbol{K} = (\frac{1}{M_A} + \frac{1}{M_B})\boldsymbol{I} - (\tilde{\boldsymbol{r}}_A\boldsymbol{I_A}^{-1}\tilde{\boldsymbol{r}}_A + \tilde{\boldsymbol{r}}_B\boldsymbol{I_B}^{-1}\tilde{\boldsymbol{r}}_B)$, wrap computed quantities as Λ_c.
Prepare a priority queue for all contact points with negative relative normal velocity, sorted by the minimum value. $\Lambda^{\min V} \coloneqq \{\Lambda_c,\ \text{key} = u_n\}$.
while $\Lambda^{\min V}$ is not empty **do**
    /* Compression Phase */
    while $\Lambda^{\min V}$ is not empty **do**
        Generate a list Γ^V containing Λ_c with minimum relative normal velocity u_n^{min} and those with similar magnitude, $\frac{|u_n' - u_n^{min}|}{|u_n^{min}|} \leq \beta$.
        Obtain increment of relative normal velocity: $\Delta u_n^{\min V} = \frac{u_n^{min}}{\alpha \cdot |\Gamma^V|}$.
        for all contact points in Γ^V **do**
            $\tilde{\boldsymbol{P}} = \Delta u_n^{\min V}\boldsymbol{K}^{-1}\tilde{\boldsymbol{n}}$, $\tilde{\boldsymbol{P}}_n = \tilde{\boldsymbol{n}}(\tilde{\boldsymbol{P}} \cdot \tilde{\boldsymbol{n}})$ and $\tilde{\boldsymbol{P}}_t = \tilde{\boldsymbol{P}} - \tilde{\boldsymbol{P}}_n$.
            if $|\tilde{\boldsymbol{P}}_t| \geq \mu|\tilde{\boldsymbol{P}}_n|$ **then**
                $|\tilde{\boldsymbol{P}}_n| = \frac{\Delta u_n^{\min V}}{\tilde{\boldsymbol{n}} \cdot [\boldsymbol{K}(\tilde{\boldsymbol{n}} + \mu\tilde{\boldsymbol{t}})]}$
                $\tilde{\boldsymbol{P}} = |\tilde{\boldsymbol{P}}_n|\tilde{\boldsymbol{n}} + \mu|\tilde{\boldsymbol{P}}_n|\tilde{\boldsymbol{t}}$
            Calculate change in velocity, $\Delta\tilde{\boldsymbol{u}} = M^{-1}\tilde{\boldsymbol{P}}$, $\Delta\tilde{\boldsymbol{\omega}} = \boldsymbol{I}^{-1}(\boldsymbol{r} \times \tilde{\boldsymbol{P}})$.
            Restore elastic energy, $\Delta W = \frac{1}{2}(2u_n + \Delta u_n^{\min V})|\tilde{\boldsymbol{P}}_n|$.
        Compute updated relative normal velocities and reset $\Lambda^{\min V}$.
    /* Restitution Phase */
    for all contact points in the contact island **do**
        Multiply restored elastic energy by Stronge's coefficient to imitate energy dissipation. $W \leftarrow \epsilon W$.

/* Separation Phase */
Build a priority queue for all contact points with non-negative elastic energy, sorted by the maximum value. $\Lambda^{\max W} \coloneqq \{\Lambda_c,\ \text{key} = W\}$.
while $\Lambda^{\max W}$ is not empty **do**
    Generate a list Γ^W containing Λ_c with maximum restored elastic energy W^{max} and those with similar magnitude, $\frac{|W' - W^{max}|}{|W^{max}|} \leq \beta$.
    Obtain change of relative normal velocity at point with largest elastic energy. $\Delta u_n^{\max W} = -u_n^{\max W} + \sqrt{(u_n^{\max W})^2 + \tilde{\boldsymbol{n}} \cdot (\boldsymbol{K}\tilde{\boldsymbol{n}})}$
    for all contact points in Γ^W **do**
        $\tilde{\boldsymbol{P}} = \Delta u_n^{\min V} \boldsymbol{K}^{-1}\tilde{\boldsymbol{n}}$, $\tilde{\boldsymbol{P}}_n = \tilde{\boldsymbol{n}}(\tilde{\boldsymbol{P}} \cdot \tilde{\boldsymbol{n}})$ and $\tilde{\boldsymbol{P}}_t = \tilde{\boldsymbol{P}} - \tilde{\boldsymbol{P}}_n$.
        if $|\tilde{\boldsymbol{P}}_t| \geq \mu |\tilde{\boldsymbol{P}}_n|$ **then**
            $|\tilde{\boldsymbol{P}}_n| = \frac{\Delta u_n^{\max W}}{\tilde{\boldsymbol{n}} \cdot [\boldsymbol{K}(\tilde{\boldsymbol{n}} + \mu \tilde{\boldsymbol{t}})]}$
            $\tilde{\boldsymbol{P}} = |\tilde{\boldsymbol{P}}_n|\tilde{\boldsymbol{n}} + \mu |\tilde{\boldsymbol{P}}_n|\tilde{\boldsymbol{t}}$
        Calculate change in velocity, $\Delta\tilde{\boldsymbol{u}} = M^{-1}\tilde{\boldsymbol{P}}$, $\Delta\tilde{\boldsymbol{\omega}} = \boldsymbol{I}^{-1}(\boldsymbol{r} \times \tilde{\boldsymbol{P}})$.
        Release elastic energy, $\Delta W = -\frac{1}{2}(2u_n + \Delta u_n^{\max W})|\tilde{\boldsymbol{P}}_n|$.
    Compute updated restored elastic energy and reset $\Lambda^{\max W}$.
/* some velocities can be negative after above two iterations */
Identify all contact points with negative relative normal velocity and update priority queue $\Lambda^{\min V}$, $\Lambda^{\min V} \coloneqq \{\Lambda_c,\ \text{key} = u_n\}$.

The kinetic energy gradually decreases to zero as a result of application of a series of impulses in the compression phase, and the relative velocity reverts to its initial sign during the energy release phase. Following Stronge's hypothesis (setting the coefficient of restitution to 0.5), the restored elastic energy abruptly decreases at the end of the compression phase and gradually decreases to zero during the energy release phase. Although the relative velocity or elastic energy converge arbitrarily close to zero, they do not achieve it numerically. Rather than that, a threshold value is chosen to truncate small values and to make the algorithm more efficient. The threshold value should be less than the timestep, as otherwise velocity changes may not be reflected, accumulating errors.

5.3.4 Time Stepping Algorithm

Explicit time integration is ideally suited to dynamic contact/impact problems, as a small time step allows for the handling of contact/impact discontinuities. They are frequently used to simulate discrete systems because the explicit methods are robust and simple to implement; they allow element-by-element evaluation and they do not require a global stiffness matrix for a discrete system, making them appropriate for initial value problem like DEM. In a traditional penalty-based DEM, the governing equations determine the resultant force acting on objects and then update accelerations and velocities via a numerical integrator. In comparison, the impulse-based dynamic simulation modifies velocity directly using a dif-

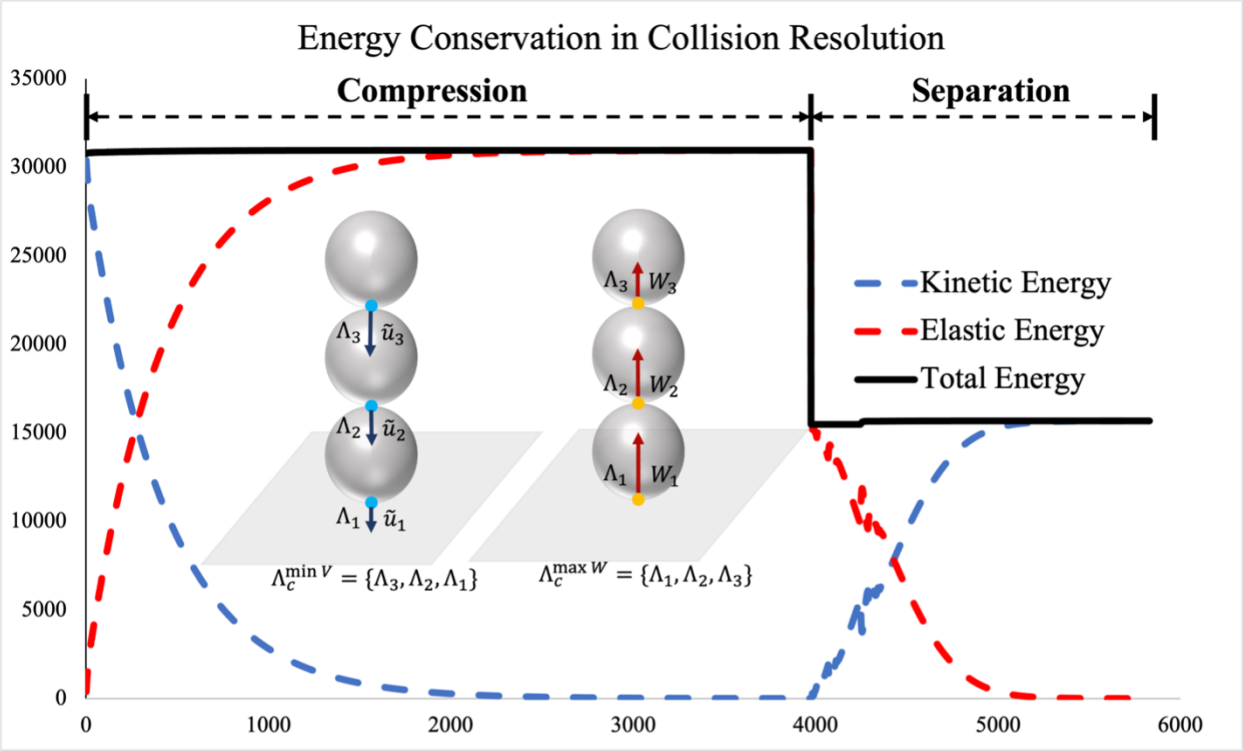

Figure 5.2: Energy conservative property of impulse-based collusion resolution, kinetic energy (blue) and elastic energy (red) converts to each other compression and separation phases. The coefficient of restitution is 0.5.

ferent numerical scheme, obviating the requirement for force computations. This results in the symplectic Euler scheme of first order.

$$\mathbf{v}^{t+1} = \dot{\mathbf{x}}^t + \Delta\dot{\mathbf{x}}^t = \mathbf{v}^t + \Delta\mathbf{v}^t \tag{5.37}$$

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \mathbf{v}^{t+1}\Delta t \tag{5.38}$$

$$\boldsymbol{\omega}^{t+1} = \dot{\boldsymbol{\theta}}^t + \Delta\dot{\boldsymbol{\theta}}^t = \boldsymbol{\omega}^t + \Delta\boldsymbol{\omega}^t \tag{5.39}$$

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t + \boldsymbol{\omega}^{t+1}\Delta t \tag{5.40}$$

Above formulations are different from the time integration methods used in penalty-based DEM, an example of second order finite difference scheme in the time-centered form (Walton & Braun, 1993) is displayed below.

$$\mathbf{v}^{t+\frac{1}{2}} = \mathbf{v}^{t-\frac{1}{2}} + \dot{\mathbf{v}}^t \Delta t \tag{5.41}$$

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \mathbf{v}^{t+\frac{1}{2}} \Delta t \tag{5.42}$$

Which is equivalent to the formulation of Lee and Hashash (2015):

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \dot{\mathbf{x}}^t \Delta t + \frac{1}{2} \ddot{\mathbf{x}}^t \Delta t^2 \tag{5.43}$$

For motion update, the penalty-based DEM is second order accurate, but the impulse-based dynamic simulation updates the motion via a linear change in velocities. In other words, the symplectic Euler scheme analyzes velocity changes in the 'secant' direction, whereas the time-centered Euler scheme considers velocity changes in the 'tangent' direction. Regarding the numerical stability, both schemes are conditionally stable and require a critical time step such that there is limited oscillation in the solution and any numerical error does not build up whereby the computed solution stays close to the truth (Lee & Hashash, 2015). Apart from that, being a Hamiltonian mechanics, the rigid body dynamic simulation also requires the numerical method does not produce spurious energy gain in the modeled system for a sufficiently long simulation time.

The symplectic Euler integration is also conditionally stable in a sense that the numerical results are bounded only if a small Δt is used that is less than Δt_{cr}. However, because the contact stiffness is omitted from the impulse-based dynamic formulation, the critical time step is determined by the Courant-Friedrichs-Lewy (CFL) condition, which is substantially larger than the penalty-based DEM formulation. The CFL condition prevents obvious penetration errors between objects, which implies that the timestep is limited by the physics of the problem, where an excessively large time step size results in objects passing through one another, but not by numerical stability issues. Additionally, it possesses the virtue of numerical stability over a long simulation period due to the fact that the total energy of a modeled system is nearly conserved even at large Δt. Indeed, an impulse-based technique may use a time step several orders greater than penalty-based DEM simulations, maintaining comparable computational fidelity. Additionally, the symplectic integrator is superior to other numerical integrators in that it is designed to preserve phase space regions even for very large timesteps (Haier et al., 2006). This feature is especially advantageous for modeling naturally deposited granular material with a large grain size distribution, for which standard DEM is prohibitively expensive. While mass scaling is widely employed in DEM (Thornton, 2000), it is not advisable if a dynamic analysis requires high frequency response.

Energy conservation is ensured by the combined usage of the symplectic scheme and iterative collision resolution algorithm. This trait, however, is less desirable when modeling a granular system with a broad variety of shapes and sizes. Consider a small grain sandwiched between two considerably larger grains, both of which are attempting to crush the small grain at opposing contact locations (Figure 5.3 left). The small grain may elastically bounce back and forth in a protracted succession of near-simultaneous encounters. This procedure

is unstable in the long run and brings the simulation to a halt. Indeed, even a very small numerical inaccuracy is likely to accumulate and result in the simulation becoming unstable. To address this issue, global damping is introduced, and the velocity change caused by global damping is updated in the following manner:

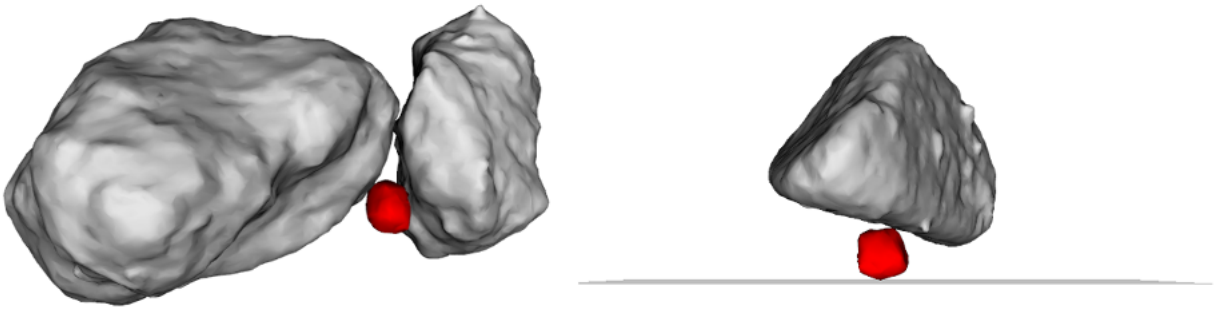

Figure 5.3: Left: A small grain is sandwiched by two larger grains. Right: A small grain is resting on rigid plane while colliding against a large grain. The collision resolution algorithm for these cases might take extremely large number of iterations.

$$\dot{\mathbf{x}} = \dot{\mathbf{x}}(1 - \xi\Delta t) \tag{5.44}$$

$$\dot{\boldsymbol{\theta}} = \dot{\boldsymbol{\theta}}(1 - \xi\Delta t) \tag{5.45}$$

$$0 \leq \xi\Delta t \leq 1 \tag{5.46}$$

Where ξ is the global damping ratio. While this approach helps accelerate convergence in the case of multiple collisions in a complicated system, it may still fail when a stiff boundary is involved. Rather being squashed by larger grains, for example, a small grain may rest on a rigid plane in one direction while colliding with a larger grain in the opposite direction (Figure 5.3 right). The rationale for applying damping to dynamic collisions between grains is that collisions will eventually resolve after a sufficient number of repetitions, and damping is a last choice for expediting this process. Interacting with a rigid boundary, on the other hand, is fundamentally different, as the rigid boundary's velocity cannot be dampened, resulting in a numerically stiff problem as discussed next.

5.4 Numerical Challenges with Impulse-Based LS-DEM

5.4.1 Contact with Rigid Boundaries

The conventional approach for impulse-based DEM is to integrate the equations for rigid body evolution forward in time and then resolve collisions to update velocities. This requires the ETM algorithm to modify the velocity discontinuously and to treat all types of interactions in the same manner to reap the algorithmic benefits. That is, regardless of the magnitude of relative velocity and friction status, both inter-grain and wall-grain interactions are considered equivalent. We will demonstrate how resting contact on rigid wall causes the ETM algorithm to enter an infinite loop when dealing with a system of complex shaped objects. We define the following terms: resting contact refers to the state of objects being stationary, whereas dynamic collision refers to the state of grains experiencing a change in force due to interference. A simplified explanation is that because the velocity of a rigid wall remains unchanged, it violates the fundamentality of the ETM algorithm, which separates two objects by sequentially applying repulsive impulses to both. Collisions require impulses to modify the velocity, whereas contacts are more closely associated with forces and accelerations. As a result, a special treatment is required to distinguish impulse-based collision treatment and needs a penalty-springs approach for at rest contact.

We used a time sequence similar to Guendelman et al. (2003) to distinguish the two types of contacts with the magnitude of the relative velocity suggested by Moore and Wilhelms (1988) and Mirtich and Canny (1995). The critical concept behind this procedure is to detect static contacts that violate impenetrability constraints and correct velocity immediately after the ETM algorithm is used to update the velocity. To avoid an infinite loop during the collision resolution, only dynamic collisions are considered. The general application of velocity correction is dependent on two critical factors: a low relative velocity and rigid boundary interaction. If one of these two criteria is not met, a wall-grain contact is still classified as a normal collision and treated using a collision resolution algorithm. After the ETM algorithm is completed, the object is advanced in the next time step using the newly solved velocities. The interference between rigid walls and grain is then checked, and the grain velocity is corrected, as we want the objects to be moved to positions where there are no rigid wall interactions. To ensure this, we use old velocities to predict the positions of rigid bodies and use corrected velocities to progress through time steps. For instance, in the collision phase, if an object's current position and velocity are $\mathbf{x}$ and $\mathbf{v}$, we test for interference with rigid walls using the predicted position $\mathbf{x}^{'} = \mathbf{x} + \mathbf{v} \cdot \Delta t$, and then apply correction to the current velocity such that the resulting velocity $\mathbf{v}^*$ does not interfere with rigid walls. Finally, we advance the object's position $\mathbf{x}^{t+1} = \mathbf{x}^t + \mathbf{v}^* \cdot \Delta t$. The algorithm's overall structure is as shown in Figure 5.4: it moves all rigid bodies to their predicted locations first, and then it identifies and resolves all grains that penetrate rigid boundaries. This idea is visualized in Figure 5.5, i.e. we consider all discretized nodes within the interpenetrating edges for a

grain and move the one with deepest penetration till it is no longer in contact with the rigid wall, angular velocity is also corrected by integrating the repulsive forces due to temporary grain-wall overlaps. After velocity correction, new contacts with negative relative velocities are identified and included in contact islands; we then re-evolve the position using the new post-collision velocity and locate grains penetrating with rigid walls once again. We repeat the process until all contacts are either non-interpenetrating or separating. By design, this time sequence also ensures impenetrability between a grain and the rigid wall; if the velocity correction step occurs before collision resolution, the objects would otherwise pass through the rigid wall. We notice that, Wriggers and Laursen (2006) and Zohdi (2014) implemented a similar adaptive time-stepping algorithm with convergence criterion to resolve the interaction between the network fabric and the rigid body.

5.4.2 Modeling Deformable Structures

In comparison to rigid bodies, dynamic simulation excludes a wide variety of methods for modeling deformable structures, as the motion of a deformable structure is frequently unpredictable from a mathematical formula, it is not considered in a dynamic simulation. As a result, impulse-based methods are frequently ignored or avoided in dynamic simulations of deformable structures, and are instead used exclusively for rigid bodies. The assertion that impulse-based methods can be applied only to rigid body models (Mirtich, 1996) is intuitive: impulses are instantaneous changes in momentum, whereas deformation is a gradual process that occurs over time. However, many applications of DEM, such as numerical modeling of experimental triaxial tests or studies of the interaction between retention barriers and boulders in debris flow simulation, require the modeling of a flexible membrane. The central idea is to represent structures using a group of rigid balls: while each ball changes velocity in a sequence of instantaneous impulses, the configuration of the balls in the group changes in a seemingly gradual manner over time. This is identical to the way we modeled flexible membrane in numerical triaxial compression and we claim that this design does not contradict to the rigid body assumption of impulse-based formulation because individual balls are still rigid. This method is different from the soft-contact model used for penalty-based DEM in that the springs used in contact detection and force resolution in penalty-based DEMs are inserted temporarily between detached colliding objects, whereas the springs used in the flexible membrane are permanent, and the distinct balls represent an entire entity. With a similar idea, (Zohdi, 2014) computationally simulated a network of coated yarn using coupled fiber-segments, which include damage and plastification of the yarn.

To demonstrate the of flexible membrane model, a representative example of several complex shaped grains falling into a flexible net is considered. The flexible nature of the net is modeled as a blanket of linked balls organized in hexagonal patterns with a stiffness of $K = 0.1$ for the internal springs between balls. The grains are subjected to vertical body force and the time step is $t = 2 \times 10^{-4}$; the simulation takes approximately $16,000$ iterations to reach an equilibrium state equivalent to 3.2 physical seconds. Figure 5.6 depicts the simulation results after 0 iteration, $4,000$ iterations, $8,000$ iterations, and $12,000$ iterations.

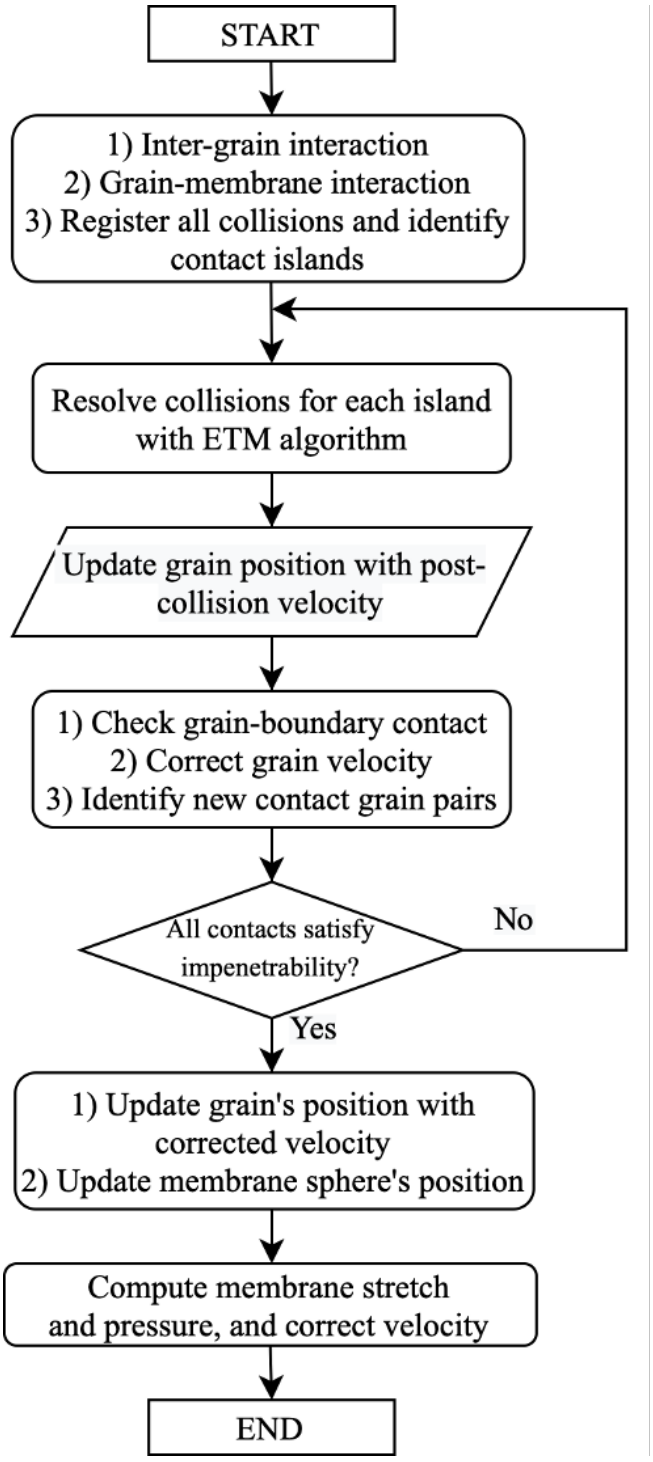

Figure 5.4: Flowchart of proposed time stepping integration, post-collision velocities were corrected by checking grain-boundary contact, membrane sphere's velocity was corrected after it advanced to the next time step to ensure impenetrability.

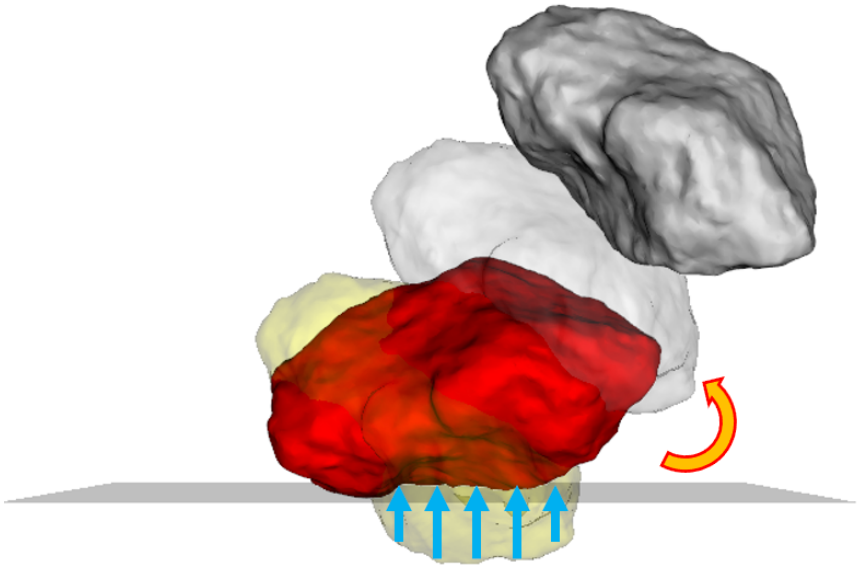

Figure 5.5: Demonstration of proposed time sequence, gray grain obtains post-collision velocity, temporarily translates to the position marked in yellow, subject to repulsive forces and correct velocities to satisfy impenetrability constraints.

It demonstrates a square flexible membrane or a net with four clamped edges. All balls along the clamped edges are given an initial velocity of zero, and forces are zeroed out, effectively immobilizing them. External downward forces are applied to grains at regular time intervals via impulses and the grains begin to fall and contact the initially flat membrane. When the membrane comes into contact, it begins to sag and then exhibits a wave-like pattern as a result of internal velocity propagation triggered by the initial fall via a chain of springs. After $12,000$ iterations, the induced waves have mostly subsided, and the grains have settled to form a depression in the membrane. This simulation took only two minutes to complete, which is compelling because it eliminated the need to select regular time intervals. The selection of an appropriate time interval in the case of granular systems is complicated by the fact that interacting entities can differ in size and momentum by many orders of magnitude. Because the size and mass of the membrane ball are much smaller than grains in this example, it would govern the time step in the conventional penalty-based DEM for the sake of numerical stability, necessitating significantly more iterations to complete the simulation.

Another numerical example is illustrated in Figure 5.7, where an assembly of grains is initially seated on an inclined plane and then subjected to gravity force. Inter-grain collisions are first resolved using the ETM algorithm, and then grain position is temporarily updated using our new time stepping scheme with the post-collision velocities. When grains collide with a rigid plane as the result of gravity and interference, post-collision velocities

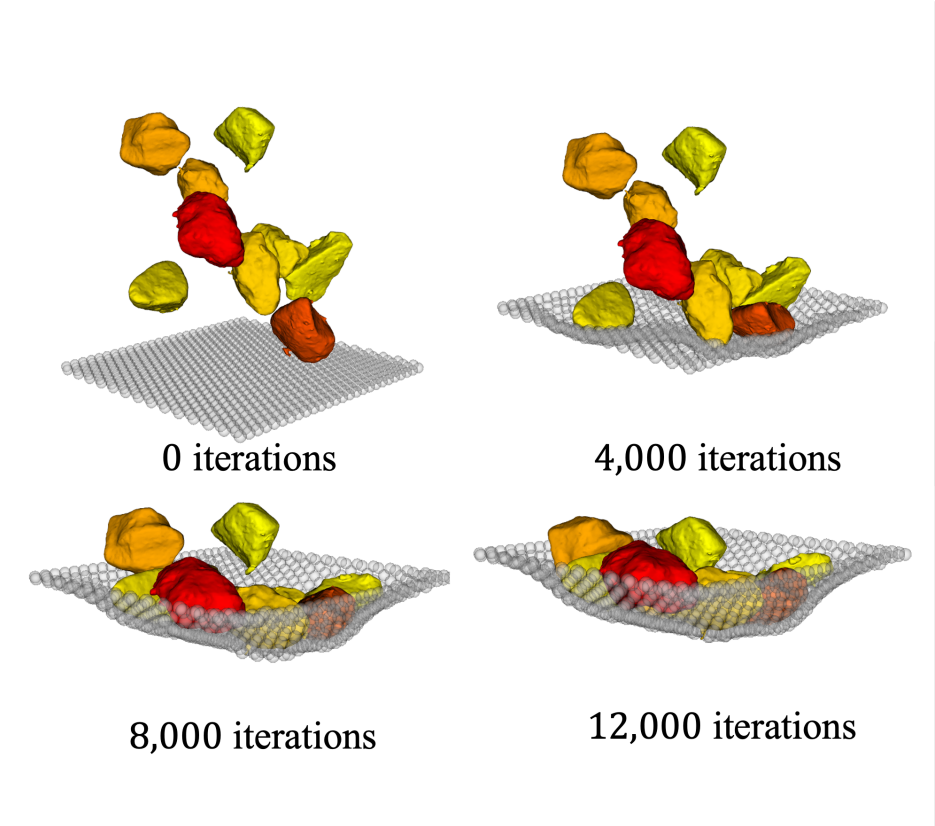

Figure 5.6: modeling of complex shaped grains falling into a flexible net.

are corrected to enforce impenetrability constraints. Then, additional pairs of grains that may collide at new velocities are identified and resolved with the ETM algorithm. When the assembly descends and rolls into the protection net, the membrane spheres interact with the grains, registering and resolving their collisions using the ETM algorithm. The membrane spheres update their positions after evolving velocities to avoid colliding with other spheres and grains. Following that, the velocity of the sphere is altered to account for spring and pressure forces acting on the membrane's surface. The sphere velocity must be corrected immediately following position advancement via post-collision velocity, and the order is critical, as the grain may otherwise pass through the membrane. A network of membrane spheres propagates a velocity wave and causes the entire entity to appear deformable in the presence of sequential collisions. Interestingly, some small grains pass through the protection net due to the stretched spheres creating gaps among them. It is worth noting that the domain re-decomposition strategy improves the efficiency of parallelized code by ensuring that the entire computational domain only encompasses the assembly's geometric configuration and distributes workload evenly across all processors. The mass difference is greater than 400 in this simulation and it does not cause the algorithm to stall, due to the fact that this is a dynamic problem which results in fewer resting contacts.

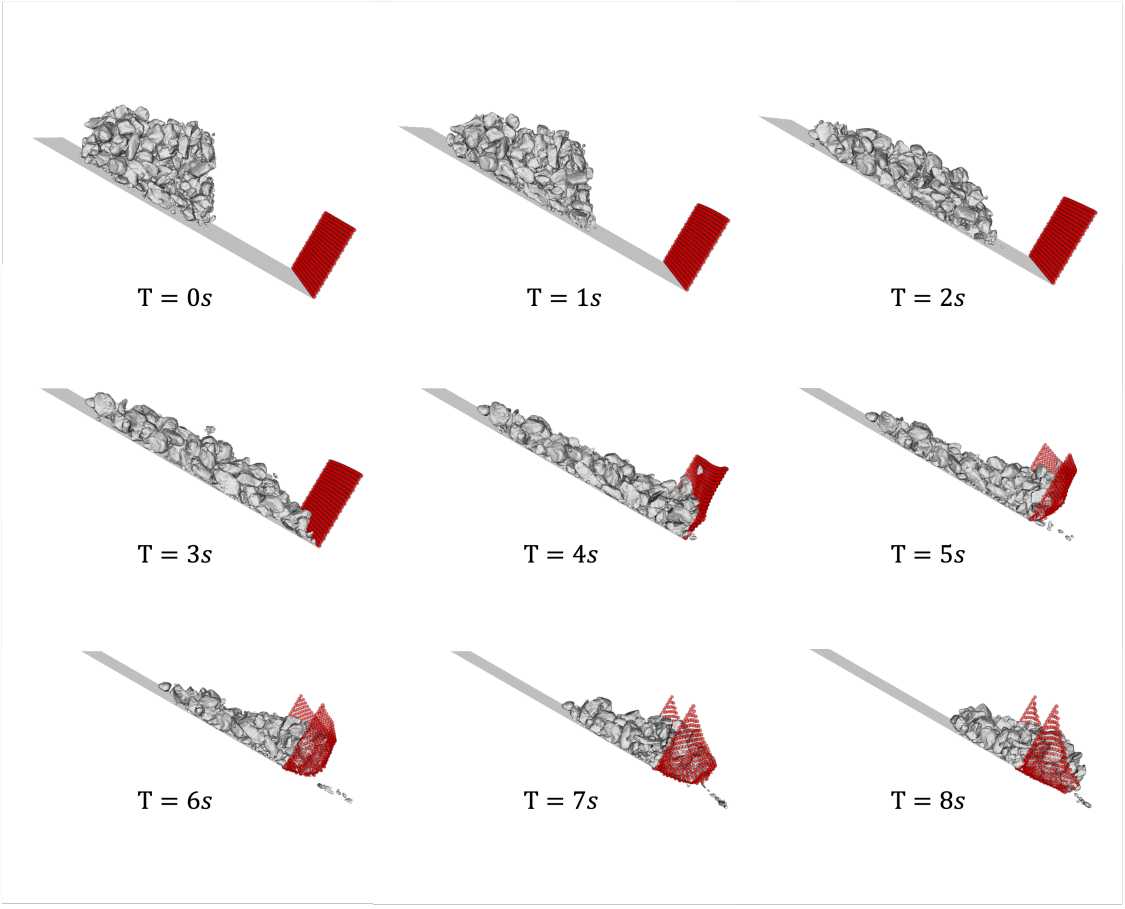

Figure 5.7: Demonstration of proposed time stepping scheme to handle rigid boundary-grain interaction and modeling of flexible membrane in impulse-based LS-DEM.

5.4.3 Use of Weighted Relative Velocity

To our knowledge, the impulse-based approach is generally used to model similar-sized objects or decompose large objects into uniform elements (Asai et al., 2021), implying that the collision matrix between attached objects is of the same order of magnitude. This does not hold for our application, as we attempted to simulate a granular system of naturally deposited sand with a wide range of grain shape and size, resulting in a colliding pair having significantly different masses. Indeed, this may introduce numerical oscillation, causing small grains to be bounced back and forth unnecessarily due to their greater sensitivity to applied impulses. To address this issue, we modified the original ETM algorithm by considering weighted relative normal velocity, defined as: $\tilde{\mathbf{u}} = \frac{m_1 m_2}{m_1+m_2}\mathbf{u}$, where m_1 and m_2 are the masses of the colliding pairs. In comparison to smaller grains. In this way, the contact points between larger grains are resolved earlier. This approach is superior to simply relying on relative velocity to determine the sequence of contacts, because larger grains respond more

gently to impulses, whereas tiny grains tend to oscillate violently even with insignificant amounts of impulses. This change enables larger grains to achieve desired post-collision velocities more quickly, and when smaller grains take control of the algorithm, the update has a negligible effect on larger grains.

5.4.4 Poorly Reconstructed Avatars

There are several issues with low-resolution grain reconstruction that may contribute to the ETM algorithm's inability to converge demonstrated in Figure 5.8. Although we do not require grain avatars to be convex, avatars constructed from low-resolution images with excessive impurities and blurred inter-grain boundaries do not accurately capture grain morphology, i.e., two avatars are merged and form fictitious geometry like the one shown below. When another grain is trapped by this strange-shaped avatar and is subjected to opposing impulses, it results in a numerical instability. Another instance occurs when a small avatar is moved or generated within a larger one, thereby subjecting all discretized nodes to impulses. Both scenarios introduce convergence issues for the ETM algorithm because one object is subjected to impulses from opposite directions exerted by a single object; this is analogous to one object being squeezed by two rigid boundaries concurrently. As previously stated, this makes it difficult for a collision resolution algorithm to converge. Numerous numerical issues also arise during high-resolution reconstruction of particles from XRCT, primarily as a result of the preservation and capture of even the tiniest grains in high-resolution images. As a result, the mass difference between the largest and smallest grains can be as large as $1,000$, resulting in a numerically intractable system. Of course, this issue can be avoided by artificially creating avatars that do not suffer from the above mentioned artifacts.

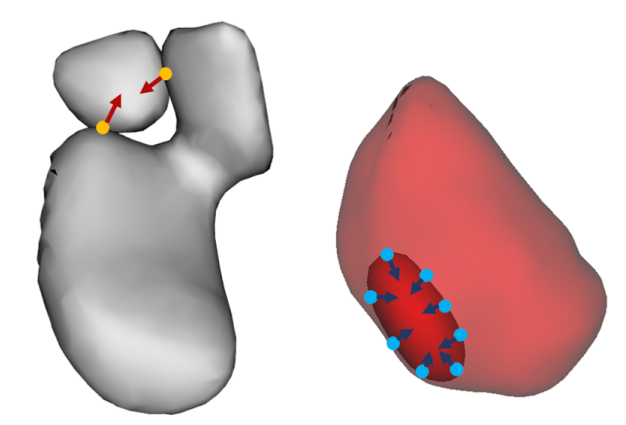

Figure 5.8: Two scenarios resulting numerical instability. Left : One avatar entrapped by another with peculiar geometry; Right : Small avatars completely wrapped inside another.

5.5 Parallel Implementation of Impulse-Based LS-DEM

The impulse-based LS-DEM continues to employ a domain decomposition strategy to improve performance. As with the penalty-based LS-DEM, the binning algorithm is implemented through a series of linked lists that map relationships between grains and bins with a search complexity of $O(1)$. There are two types of data communication to consider: border communication and grain migration. The first type is treated identically to the penalty-based LS-DEM, and an efficient algorithm consisting of three sequential calls to the optimally tuned MPI routine MPI_Sendrecv is sufficient to update exchange halo layers and automatically handle edge cases. Grain migration across sub-domains is a complicated step in the penalty-based LS-DEM because the contact history of a migrating grain is also brought along if the history-dependent tangential contact model is used. However, it is much easier to handle in an impulse-based method because it does not track the evolution of friction forces, allowing all quantities to be packed and communicated collectively, thereby lowering communication overheads.

The domain decomposition strategy divides a large computation task into smaller tasks and assumes that each sub-domain has access to all necessary resources to run independently. This is consistent with the penalty-based method, in which all grain motions are a result of contact forces and have no relationship to distant grains. Because the collision reaction is associated with a change in the microstructure at the contact point, the change in acceleration must occur over a positive time interval. Indeed, the forces take some time to propagate throughout the body due to the body's elastic nature. Collision contact, however, occurs within a very short time interval for a rigid body, which means that displacement and change in the contact area are negligible or remain unchanged. Thus, in a penalty-based DEM simulation, a body is influenced solely by its immediate contact neighbors, and it is sufficient to consider a layer of halo region for data communication across computational sub-domains. While inter-grain interactions can be reconciled element by element using a penalty-based approach, a contact island, as illustrated in Figure 5.9, in which any pair of grains is connected via a chain of immediately contacting grains, must be solved concurrently to satisfy the impenetrability constraints in the impulse-based formulation. Thus, one of the difficulties in parallelizing impulse-based LS-DEM is that a contact island can span an arbitrary number of sub-domains, resulting in grains within a sub-domain becoming indistinguishable from those in other sub-domains. Additionally, the penalty-based method obtains boundary interactions from the halo layer and updates the grain's motion independently of other processors, whereas the impulse-based method requires one processor to gather and resolve all collisions in a contact island.

The force resolution step of impulse-based method continues to use an $O(n)$ binning algorithm. This step identifies and registers all contact points that violate impenetrability constraints, including contact IDs, branch vectors, and the normal direction of a collision between two avatars. Following that, the master MPI processor (rank 0) collects these

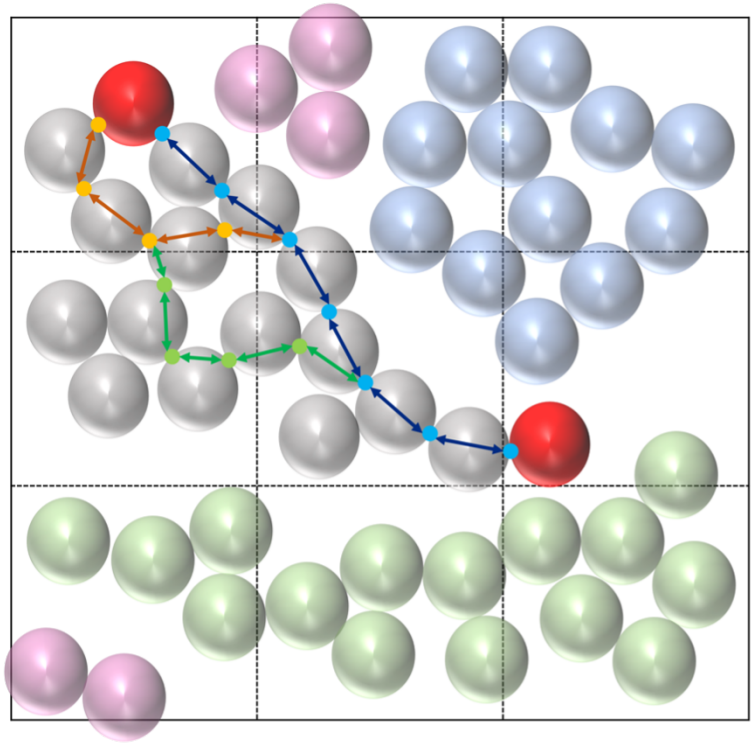

Figure 5.9: Illustration of contact islands (indicated in different colors) where a group of bodies are associated via a contacting chain which could span over several sub-domains. For example, two red bodies in a same group can influence each other possibly via several ways.

contacts and summarizes a list of contact IDs, determining the number of contact islands and grains contained within. The concept of obtaining contact islands was inspired by the fact that colliding bodies make contact with one another, and that the system of all colliding pairs forms an undirected graph. As a result, the problem of determining contact islands becomes a graph partition problem of determining all connected components. Additionally, we can consider the graph as a sparse, symmetric square matrix whose dimension equals the number of grains and whose entries are non-zero only when the corresponding pair of grains collides. When the problem is rephrased as permuting such a matrix into a band matrix with a small bandwidth, the Cuthill-Mckee algorithm (Cuthill & McKee, 1969) is used, which is based on the graph's Breath First Search (BFS) algorithm. Algorithm 4 contains the Cuthill-Mckee algorithm optimized for our application. Following the identification of contact islands, collisions between rigid bodies are resolved within each contact island, which may span multiple sub-domains. To minimize data communication, a processor gathers a contact island that already contains the majority of collisions. Due to the small and specific size of the data being communicated, it can be packed and transferred efficiently and collectively. After collision resolution is complete, the change in velocity of each grain is computed and distributed back to the grain's original process, which updates the grain's

velocity and advances a timestep. The preceding procedure formalizes the parallel strategy used in Algorithm 5 for impulse-based LS-DEM.

Algorithm 4 CutHill-Mckee Algorithm

Input: A list L^{ID} with length of N_{grains} including IDs of contact neighbors for all grains in assembly.

Output: the number of contact islands in an assembly N_{islands}, a list Λ^{islands} of size N_{islands} such that $\Lambda^{\text{islands}}[i]$ denotes IDs of elements in the i-th contact island.

From L^{ID} generate a list L^{degree} denoting the degree of an element, and a list L^{checked} denoting if an element is looked up already.
Set the number of contact islands $N_{\text{islands}} = 0$, the total number of checked elements $N_{\text{checked}} = 0$; and prepare a priority queue Λ^d for elements in the assembly, sorted by the minimum degree.
while $N_{\text{checked}} < N_{\text{grains}}$ **do**
    Iterate L^{degree} to find an element i with minimum degree among all unchecked elements.
    Sort all elements in $L^{\text{ID}}[i]$ into Λ^d if they are not already, and set $L^{\text{checked}}[i] = \textit{true}$.
    while Λ^d is not empty **do**
        Pop out the first element j from Λ^d.
        if j is not already in $\Lambda^{\text{islands}}[N_{\text{islands}}]$ **then**
            Insert j into $\Lambda^{\text{islands}}[N_{\text{islands}}]$, and set $L^{\text{checked}}[j] = \textit{true}$.
            Sort all elements in $L^{\text{ID}}[j]$ into Λ^d if not already.
    $N_{\text{checked}} \leftarrow N_{\text{checked}} + |\Lambda^{\text{islands}}[N_{\text{islands}}]|$
    $N_{\text{islands}} \leftarrow N_{\text{islands}} + 1$

The flowchart of parallelized impulse-based LS-DEM is shown in Figure 5.10. There are several implementation details worth attention. As the basic element of impulse-based method is an island of collisions, which could spread over multiple processors, each processor still needs to generate a simplified avatar containing mass, moment of inertia for those that do not belong to the sub-domain, and keeps track of all grains' position, rotation, velocity, and angular velocity. In addition, both post-collision velocity that is attained from solving a contact island and the corrected velocity after interacting with rigid boundaries must be broadcast to all processors before updating grain's motion. Moreover, since each processor interacts with its neighbors via halo layers, the grain-halo contacts are duplicated when contacts in an island that spreads multiple processors are gathered to one processor, hence half of interactions with halo layers need to be removed.

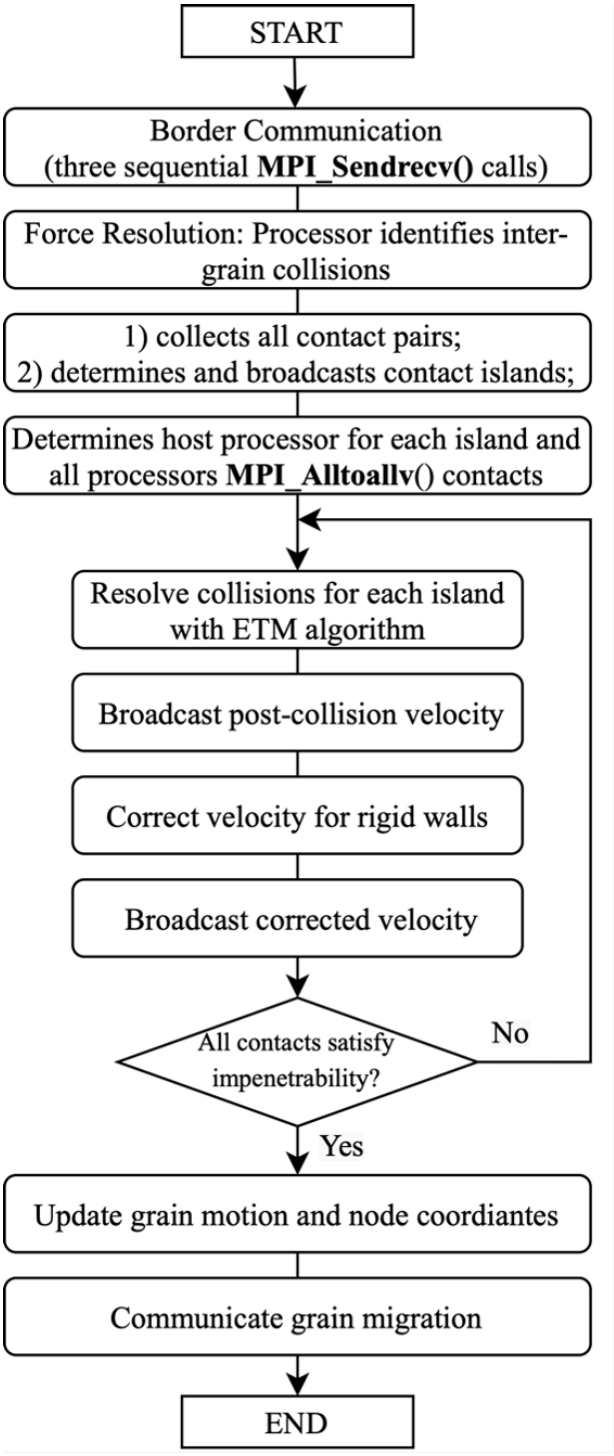

Figure 5.10: Flowchart of parallelized impulse-based method.

Algorithm 5 Algorithm to Parallel Contact Islands

Input: MPI rank i maintains a list Λ_C^i of contact details Φ_C including contact IDs of colliding bodies A and B; branch vectors from centers to contact point $\boldsymbol{r}$; contact normal direction $\tilde{\boldsymbol{n}}$; relative velocity $\tilde{\boldsymbol{u}}$; elastic energy W and body mass M, such that $\Lambda_C^i \coloneqq \{\Phi_C : \Phi_C \in \text{rank i}\}$.

/* Start Graph Partition to Detect Contact Islands */
if *rank_id* $\neq 0$ **then**
    Generate a list Λ_{ID}^i contains IDs of contacting neighbors for each grain belongs to the rank.
    MPI_Send Λ_{ID}^i to rank 0.
else
    for *rank_id* $= 1 : N_{\text{ranks}}$ **do**
        MPI_Recv Λ_{ID}^i from rank *rank_id*.
    /* Prepare For the CutHill-McKee Algorithm */
    Build a list L^{ID} with length of N_{grains} to register contact IDs for all grains in assembly.
    Prepare a list Λ^{islands} of size N_{islands} such that $\Lambda^{\text{islands}}[i]$ denotes IDs of elements in the i-th contact island.
    $[N_{\text{islands}}, \quad \Lambda^{\text{islands}}] =$ **CutHill-McKee**(L^{ID}).
    for *island_id* $= 1 : N_{\text{islands}}$ **do**
        determine the host rank that contains most collisions of the i-th contact island, and register the host ranks in list L^{host} with length of N_{islands}.
/* Broadcast Contact Islands Information */
MPI_Bcast Λ^{island} and L^{host}.
/* All-To-All Communicate Contact Detail Φ_C */
MPI_Alltoall number of contact details that each rank should receive from others.
MPI_Alltoallv specific contact details that each rank should receive from others.
/* Collision-Resolution Phase */
for *island_id* $= 1 : N_{\text{islands}}$ **do**
    if $L^{\text{host}}[\textit{island\_id}] ==$ *rank_id* **then**
        Resolve collisions via **Collision-Resolution Algorithm**.
/* Broadcast and Update Velocities */
MPI_Allgatherv linear and angular velocities from other ranks.

5.6 Numerical Tests

5.6.1 Performance Speedups with Impulse-Based LS-DEM

The speed-up of impulse-based LS-DEM was investigated by modeling the flow of 1600 arbitrarily complex shaped grains under artificial gravity from a smooth container, as shown in Figure 5.11. The grains employed in the assembly have wide range of grain shapes and sizes, making the time-step of conventional penalty-based approach very small. Table 5.1

shows model parameters for the penalty-based and the impulse-based LS-DEM. The values for inter-grains friction coefficient μ, normal contact stiffness K_n, and shear contact stiffness K_s were adopted from Kawamoto et al. (2016). The coefficient of normal restitution according to Stronge's hypothesis is taken $R_n = 0.65$, which was obtained by Li et al. (2021) by conducting experiments. The coefficient of tangential restitution R_t needs also to be calibrated, but many impulse-based simulators show good physical plausibility with a default value of zero (Lee & Hashash, 2015). Global damping is $\xi_g = 0.5$ for both approaches. Different time steps for impulse-based LS-DEM (Δt_i) were chosen to study the speed up as well as the simulation fidelity over the reference penalty-based simulation. The time step of penalty-based LS-DEM was computed by assuming a factor of safety 0.1 and the value 2.5×10^{-4} is selected for the ease of comparison.

$$\Delta t = 0.1 \times \sqrt{\frac{K_n}{M_{\min}}} \approx 2.98 \times 10^{-4} \tag{5.47}$$

Common Parameters		
Specific gravity of solids	G_s	2.65
Inter-grain friction coefficient	μ	0.55
Global damping coefficient	ξ_g	0.5

Penalty-based LS-DEM			**Impulse-based LS-DEM**		
Normal contact stiffness	K_n	3×10^4	Coefficient of normal restitution	R_n	0.65
Tangential contact stiffness	K_s	2.7×10^4			

Table 5.1: List of model parameters and values used for the simulations.

In the simulations, the outer walls of the container were removed and the sand was allowed to flow out. The geometries of the resulting mounds, approaching the angle of repose, were compared at the end of 8s time interval. In each timestep, the post-collision velocities were corrected for rigid boundary contact until the computed velocities did not violate either inter-grain or grain-wall impenetrability. In this example, we repeated this procedure several times within a single timestep, and we found 10 iterations were enough to stabilize the velocity of grains. The code speed-up was measured in terms of central processing unit (CPU) time. Five simulations using impulse-based LS-DEM were conducted, with the time step used varying from $\Delta t_i = 1\Delta t = 2.5 \times 10^{-4}$ to $\Delta t_i = 16\Delta t = 4.0 \times 10^{-3}$. Figure 5.12 shows the mound geometry for each simulation and Table 5.2 tabulates the speed up times for the different time steps in the impulse-based DEM simulations. The simulation fidelity can be checked in terms of final height and spread of mound. Although all simulations produced similar shaped mounds, discrepancies in positions and rotations of individual grain can be large. Calibrating impulse-based method to achieve reasonable agreement to reference penalty-based simulation is not easy as it requires the identification of actual or ad hoc internal variables and constitutive relations. In addition, there are too many degrees of freedom even for a single grain and to make the one-to-one comparison is

Figure 5.11: Initial configuration of 1600 very high-resolution grains in $800 \times 800 \times 800$ domain for impulse-based method with large timesteps.

not feasible. Instead, we aimed to explore to what extend that both methods agree with each other in a qualitative sense. At the same time we found that the simulation results even using a very large time steps, showed no obvious penetration errors between particles and the geometric fidelity was maintained across all simulations.

	Δt (sec)	CPU Time (mins)	Time Step Diff	Speed-up
Penalty-based	2.5×10^{-4}	$\sim 4,000$		
Impulse-based	2.5×10^{-4}	265	$1\Delta t$	15.1
	5.0×10^{-4}	132	$2\Delta t$	29.4
	1.0×10^{-3}	72	$4\Delta t$	55.6
	2.0×10^{-3}	42	$8\Delta t$	95.2
	4.0×10^{-3}	42	$16\Delta t$	95.2

Table 5.2: Comparison of CPU time and speed up of impulse-based methods.

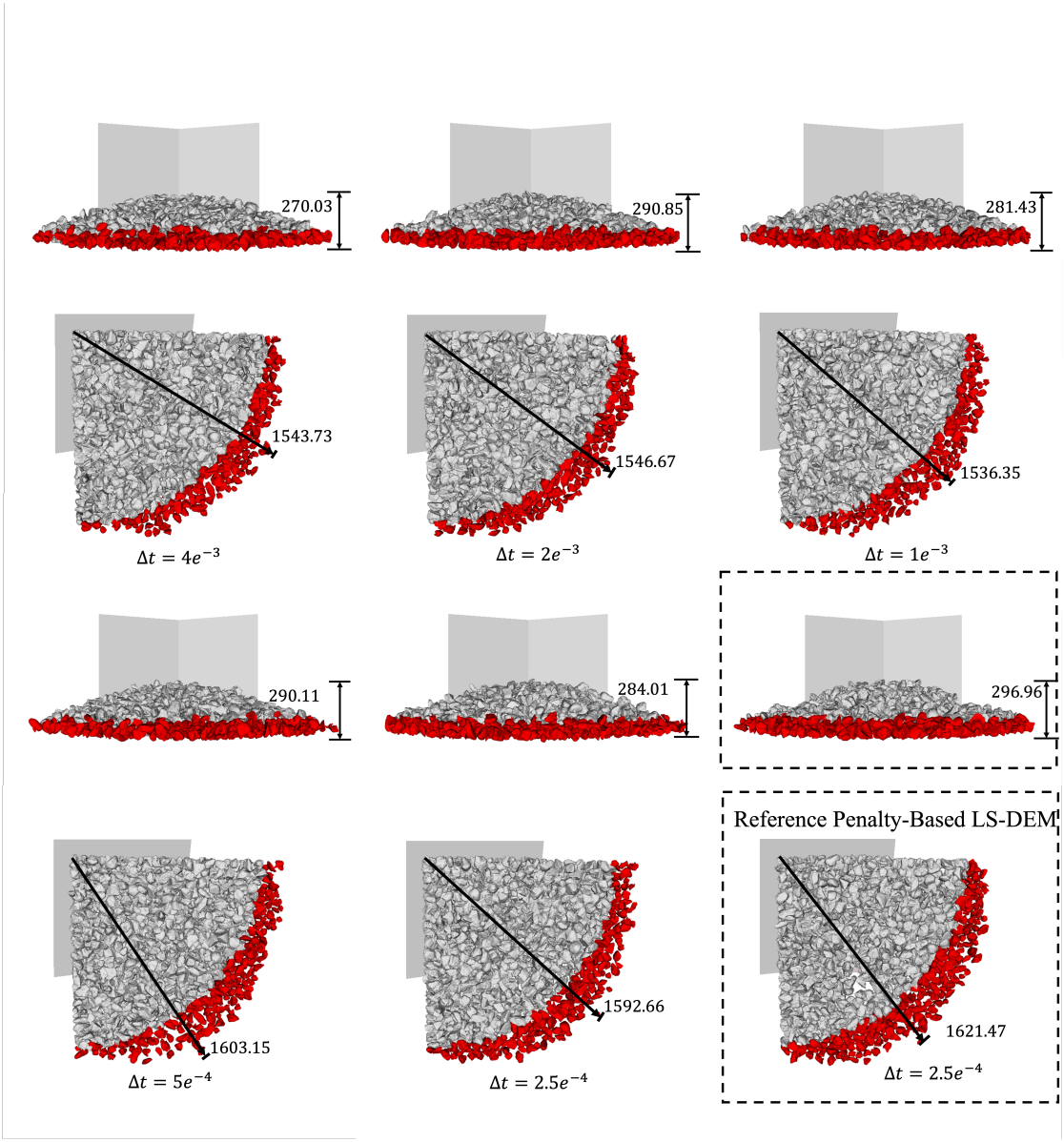

Figure 5.12: Specimen height and spread after 8s settlement under gravity with varying timesteps.

In this simulation, the particles were reconstructed from a very high-resolution dataset ($3.4\mu m$/pixel) and each grain was represented by $2,561$ discretized nodes on the average and storing $1,600$ grains required 3.36 GB of memory. We rarely used very high-resolution avatars for large scale simulation with penalty-based method although we have efficiently parallelized code, because actual workload corresponds to the total number of nodes rather than the number of grains. In contrast, there are fewer restrictions to model high-resolution

avatars with impulse-based method. Even though impulse-based method still interpolates normal direction from the LS grids, the impulse-based LS-DEM does not compute, update, and track complicated history-dependent friction. This simplified force resolution and helped to keep LS grids in cache and enabled continuous, fast access. Besides, the amount of time required for sub-iterations to resolve multiple collisions is less demanding because doing arithmetic operations is hundred times faster than moving data around in modern machines.

Significant speed up shown in Figure 5.13 is achieved in impulse-based method with larger time steps, which is not likely for conventional penalty-based DEM. As grains settled down, the number of contact islands decreased and the number of contacts in an island increased. We observed that even though as many as 300 contact islands still remained at the end of simulation, the largest number of contacts in an island was more than $2,000$, this implies most of grains were grouped into several clusters and time used to resolve collisions raised. We also found that both the number of contact islands and contacts increase with timestep as larger timestep changes configuration more rapidly and adds more collisions. The ratio of mass differences between the largest and the smallest grains in this example was around 30, and we found this is a ratio that our algorithm could run under different timesteps, accelerations and boundary conditions.

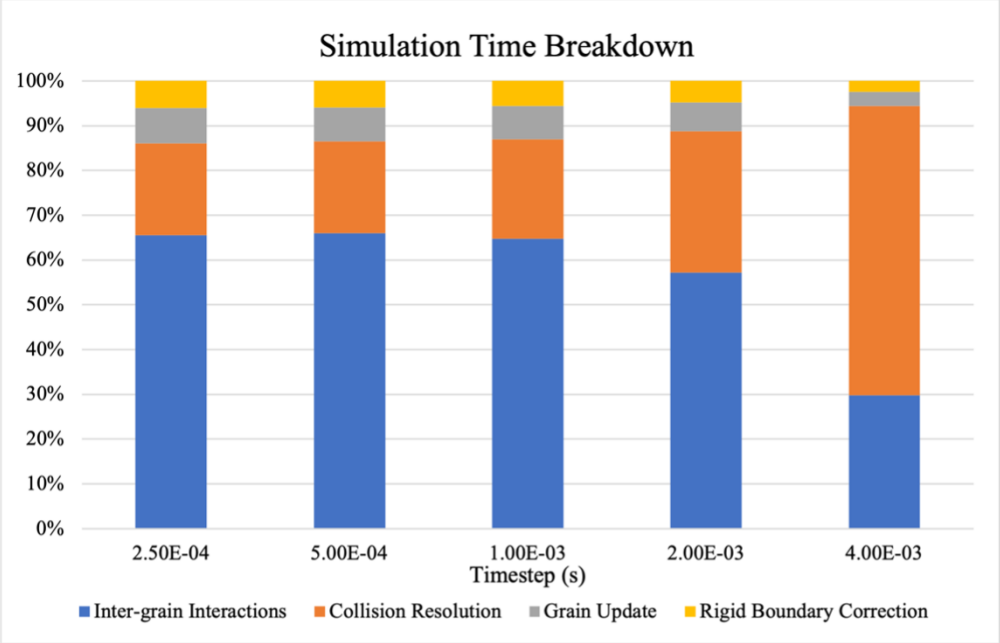

Figure 5.13: Simulation time breakdown for various timesteps.

Figure 5.14 shows why computing time for simulations using timestep 2.0×10^{-3} and 4.0×10^{-3} does not change substantially. Although none of simulation results violated CFL condition and incurred obvious penetration errors, larger timestep identified more contacts and required substantially more time to solve ETM algorithm. As shown in Figure 5.14, the

resolution time for contact islands with various number of collisions is nonlinear. This implies that, although not violating the physics of the problem whereby large timestep results in grains passing through each other, the timestep should also be chosen to avoid spending too much time in collision resolution.

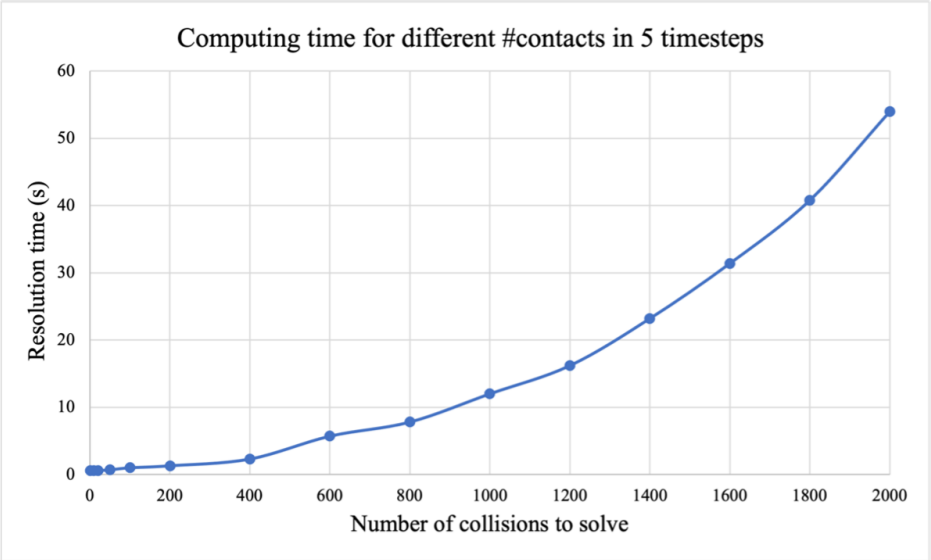

Figure 5.14: Computing time for islands having different number of collisions, all measured for 5 timesteps and 10 sub-iterations for rigid boundary correction.

5.6.2 Efficiency Analysis of Parallel Impulse-Based LS-DEM

To benchmark the performance of parallelized impulse-based LS-DEM, a series of numerical experiments were performed on the Savio system of UC Berkeley. The objective was to measure the performance improvement by integrating an $O(n)$ algorithm and using parallel techniques. We used XRCT data to reconstruct the observed sand fabric with $155,016$ avatars as shown in Figure 5.15. The reconstructed avatars were tightly packed which is not the favored type of problem for an paralleled impulse-based code for two reasons. First, the system of granular particles has wide range of masses, shapes, and sizes; second, almost every grain is connected in a single cluster and all inter-grain interactions are static contacts. These issues make the problem numerically too stiff to be modeled as an impulse dynamic problem. To tackle such limitations, we removed the avatar with mass two standard deviation away from the mean to make the mass ratio between the grains at most 30 and with 299 discretized nodes per grain on average, the model required 12.6GB memory for morphological files. The demonstrated problem is simple: domain boundaries are modeled as undeformed planes; grains are restricted from leaving the domain and would be bounced back if it intends to do so. For simplicity, grains are subjected to random accelerations in each time step to

secure load balance. The bookkeeping and adaptive binning subroutines are switched off when the performance is benchmarked. With given computing resources, the code could adaptively determine the number of processors that are actually utilized such that the total border/ghost area or the amount of communication across processors is minimized.

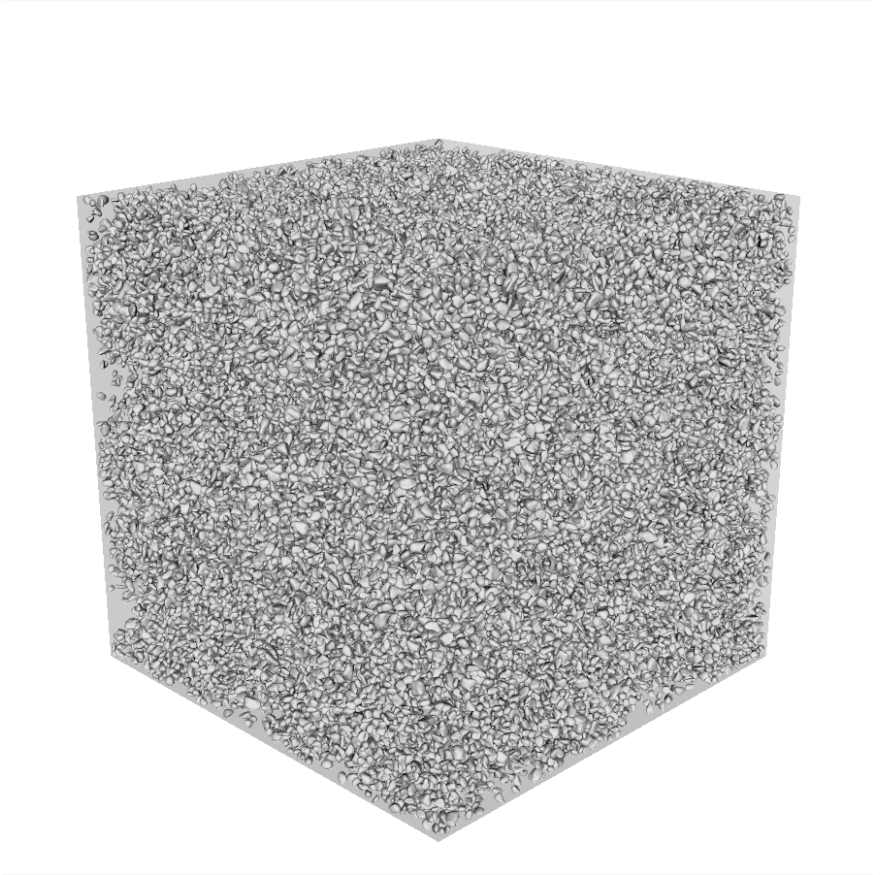

Figure 5.15: $155,016$ low-resolution grains in $1,200 \times 1,200 \times 1,200$ domain to benchmark parallel impulse-based LS-DEM.

Figure 5.16 lists the simulation time breakdown for parallelized impulse-based LS-DEM. While the serial portion of code consumed less time with more processors involved, communication overheads nevertheless began to govern and computing time for collision resolution phase soon ceased to decrease. All speed-ups were measured relative to a serial run, which is equivalent to an MPI simulation with a $1 \times 1 \times 1$ decomposition. The performance gains by using 8, 27, 64 and 216 processors are 5.5, 7.0, 7.3 and 5.0, respectively. As is evident, a parallel code cannot achieve high efficiency unless communication overheads are kept to a level, beyond which more MPI processors lead to an increased amount of MPI traffic and stall or even deteriorate scalability. In this case, the impulse-based method could not be perfectly parallelized as we achieved with the penalty-based method, but there is still a sweet

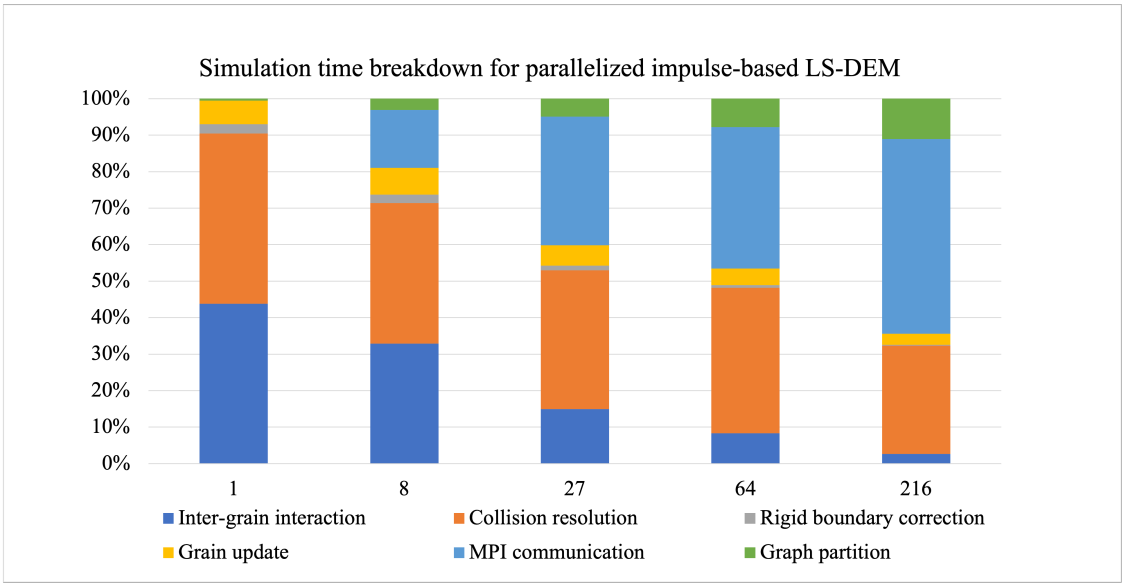

Figure 5.16: Simulation time breakdown for parallel impulse-based LS-DEM, modeled with 155, 016 grains, with 1, 8, 27, 64, 216 processors, respectively.

spot, eight processors in this specific case, where the communication is still modest, while increasing the number of processors to beyond 64 no longer brings additional benefit as the MPI communication starts overwhelming the other operations.

5.6.3 Numerical Modeling of Rock Avalanche

Even though Impulse-based DEM is capable to take much larger time step and eliminate substantial amount of demand to compute forces, the fundamental issue of a discontinuum-based method to model an assembly with wide range of object sizes still remains. This is the issue we already addressed when discussing the problem of modeling fixed, or rigid boundaries. In that context, modeling topography presents an analogous problem.

Within the LS framework, any arbitrarily shaped surface, such as a real topography with ridges and valleys, can be captured provided its associated elevation data. The LS algorithm was first used to locate the zero-valued contour, from which the fast-marching method (Sethian, 1999) is used to construct the underlying signed distance grid by solving a boundary value problem of Eikonal equation:

$$|\nabla \phi(x)| = \frac{1}{f(x)} \quad \text{for} \quad x \in \Omega \tag{5.48}$$

$$\phi(x) = 0 \quad \text{for} \quad x \in \partial\Omega \tag{5.49}$$

Such a problem describes the evolution of a surface as a function of LS ϕ with speed $f(x)$ in the normal direction at a point x on the propagating surface. Alternatively, $\phi(x)$ can be

thought of as the shortest time required to reach x starting from the zero contour, of course, $\phi(x) = 0$ for x on the zero LS contour $\partial\Omega$. The algorithm is similar to Dijkstra's algorithm and uses the fact that information only flows outward from the area that is already labelled with a value. In this example, a digitized topography (see, Figure 5.17) and its LS representation are constructed as a large LS-DEM object to exploit algorithmic advantages and we treat grain-topography interaction identically to grain-grain interaction. The digitalized topography occupies the entire computational domain and consumes as much as 10GB computer memory. To avoid large memory demand, we only save a narrow band of grid adjacent to the zero LS contours, which is sufficient to interpolate the amount of penetration between objects and keep the memory requirement low.

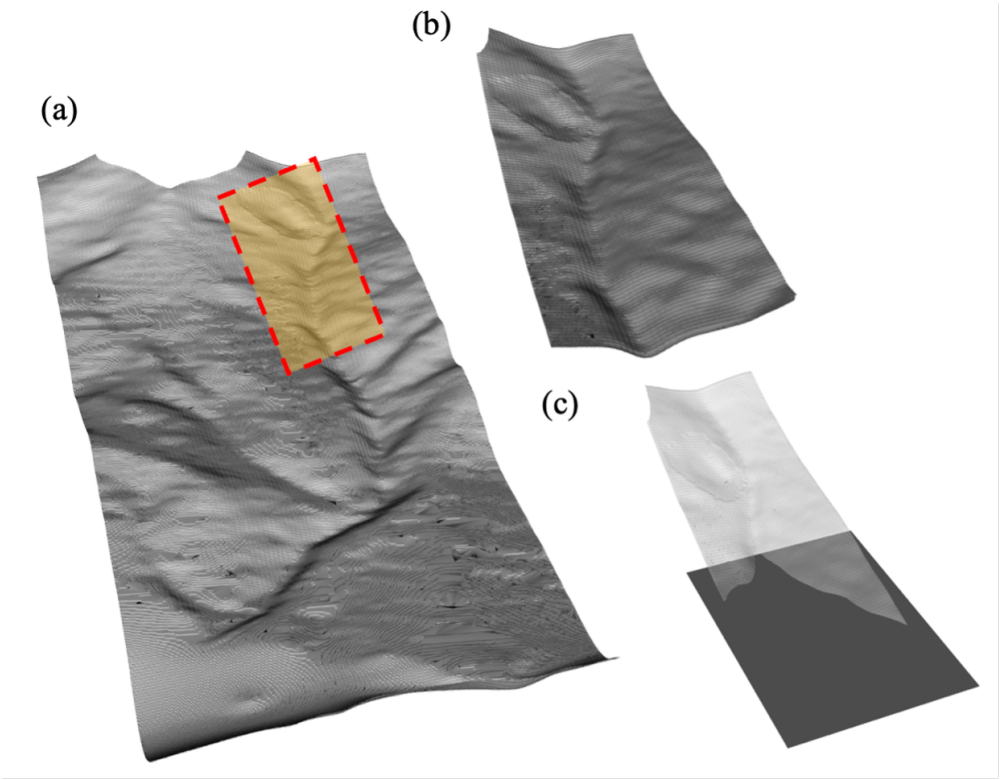

Figure 5.17: (a) Entire digitalized topography. (b) A zoomed in region marked in yellow. (c) A zoomed in region concatenated by a flat plane.

We show three simulated landslides in which a granular deposit assembly was perturbed by gravity and allowed to slide as gravity overcame the internal friction of the deposit. The digitalized topography is inclined at 25°, and the coefficient of internal friction between the particle and the topography is 0.2. Larger particles were visualized in red, while smaller particles were displayed in orange. In all cases, the particle mass, rock avalanche, moves fluid-like down the stream channel. In the first example (Figure 5.18), the avalanche ran out on a flat plane and formed fan-shaped deposit as is typical of rock avalanche deposits.

In the second example shown in Figure 5.19, the deposit ran out onto a surface sloping at only 3°, again similar to many natural settings, and was deflected downslope. In the third example (Figure 5.20 and Figure 5.21), we simulated different geometries of flexible barriers, such as are often used for rock fall protection. In both cases, the avalanche was initially contained until it overflowed the barriers. Our interest in these simulations was to explore the potential of the impulse-based DEM to realistically handle field-size problems rather than solve a particular problem. The first two simulations ran on a single CPU and completed $20,000$ steps in 2 hours, while the third simulation completed the same number of steps in 5 hours due to the more complicated membrane-grain interaction. Overall, the results are very promising, as rock avalanches have been typically modeled as equivalent viscous fluid (such as, Setiasabda, 2020; Ho et al., 2021), rather than assemblages of particles, in order to make the simulation tractable.

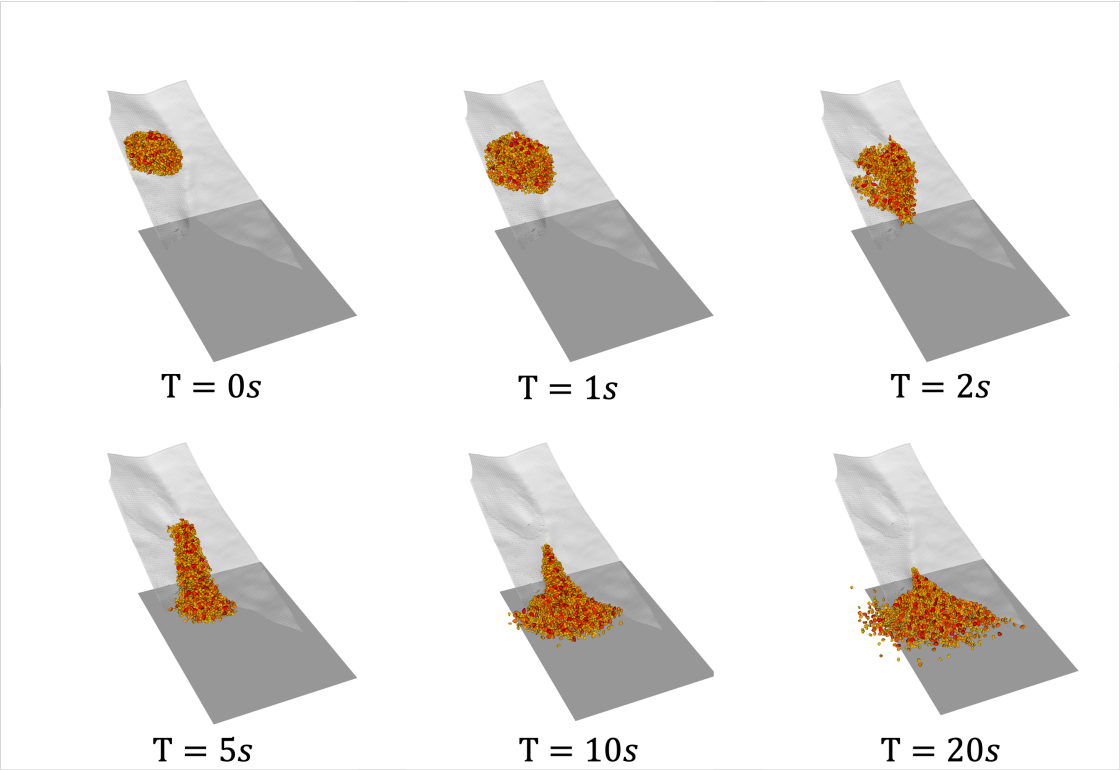

Figure 5.18: Rock avalanche runs into a flat plane.

5.7 Conclusion

We have showed that impulse-based method is compelling in large part that it alleviates the need to choose regular time intervals and saves substantial amount of force computation

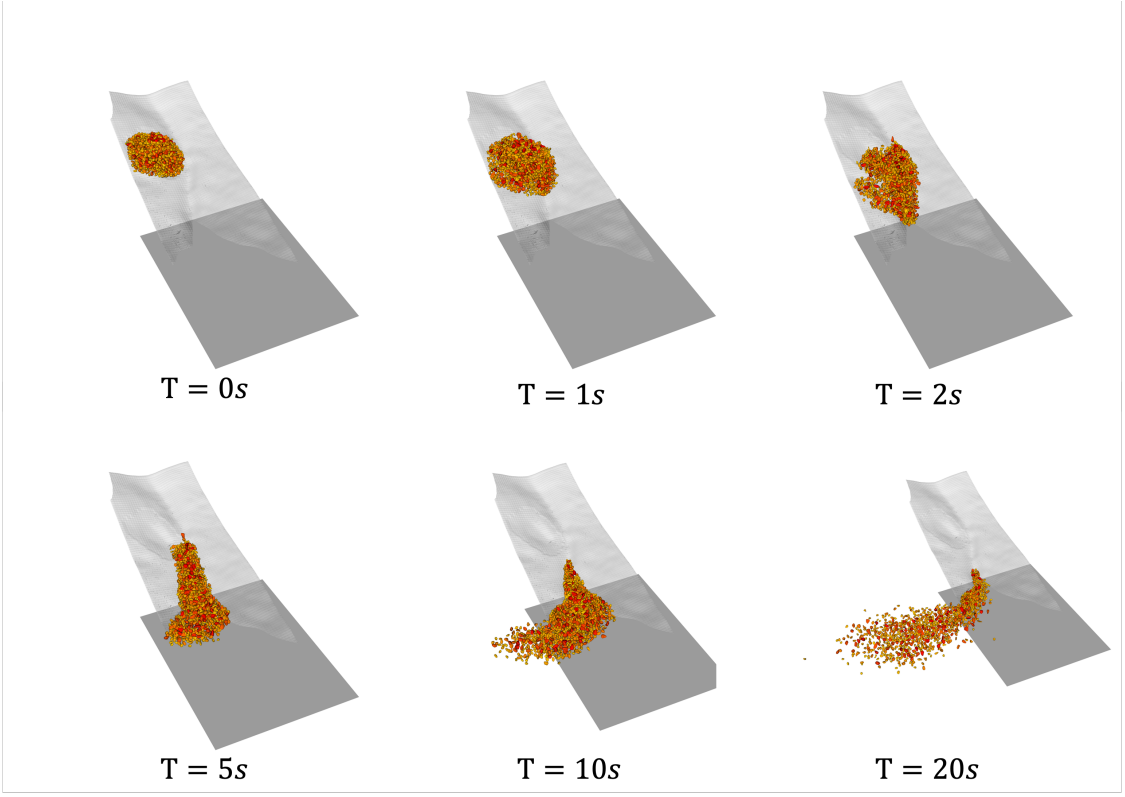

Figure 5.19: Rock avalanche runs into a tilted plane.

for complex shaped objects. The combined effects speed-up the simulation by at least factor of 10 with reasonable levels of fidelity. We modified ETM algorithm making it more suitable for dealing with a system of wide shapes and sizes, instead of removing the minimum smallest relative velocity from priority queue each time, we prioritize the weighted relative velocity which considers the masses of colliding bodies for better convergence. This measure is effective because it first considers larger objects as later updates of small objects only trivially influences the larger ones. Compared to SMM and SQM, ETM algorithm is advantageous that it is less sensitive to the order of impulses and tends to be more energy conservative. By applying impulses gradually to the contact points, the impulses can influence multiple contact points at the same time and enable propagation of impulse effects and improve the overall ability of the method to capture propagation of forces during dynamics simulations. We further incorporated impulse-based method into LS framework, the geometric representation of irregular grains with signed distance functions defined on grids. We also used a novel time integration scheme that separates the treatment of inter-grain collision and wall-grain resting contact. This improves the robustness of collision resolution algorithm, therefore avoiding unreal numerical oscillation and energy dissipation. Also, the demonstration of protection nets modeling provides convincing evidence that impulse-based methods can be used to simulate the dynamics of deformable structures.

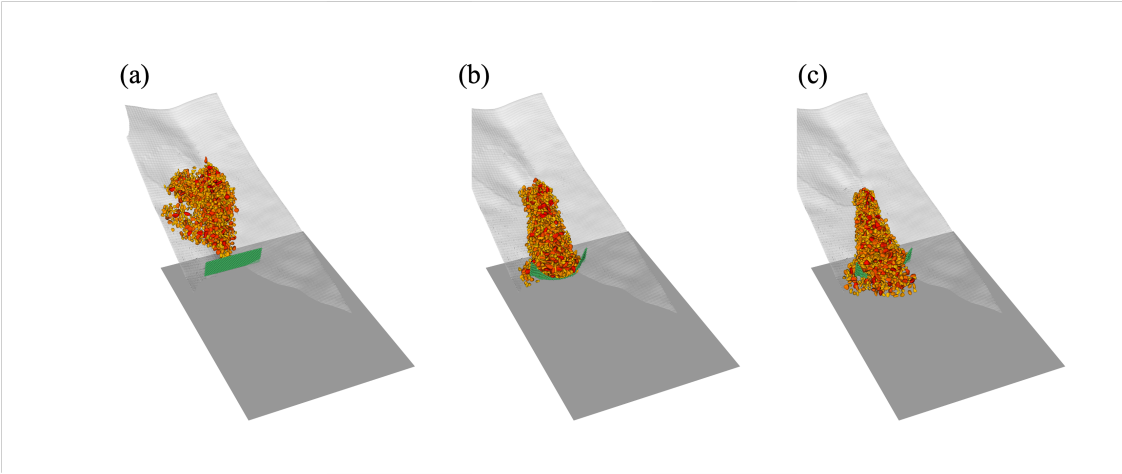

Figure 5.20: Rock avalanche impacts a single protection net.

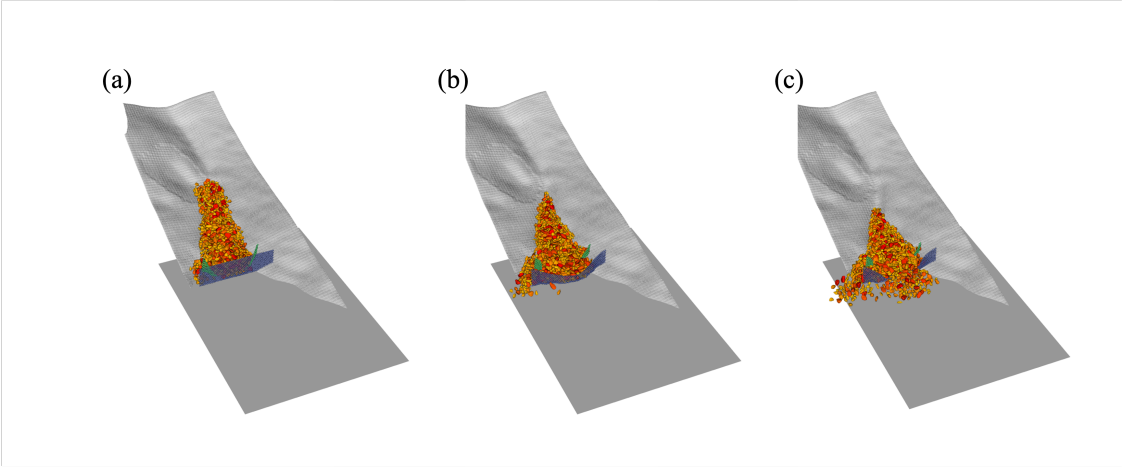

Figure 5.21: Rock avalanche impacts a double-layer protection net.

The major limitation of impulse-based method still lies in modeling a system of irregular, nonconvex, non-uniform objects especially in a highly confined quasi-static setting, whereby objects with very different shapes and sizes interact with each other at many contact points and have small velocities. However, our modified ETM algorithm and time integration scheme is capable to simulate a range of very different objects with each of them being modeled as an entity and draw least amount of artificial interference for dynamic problem as we presented. We are interested in enhancing our current simulator to make it more robust analysis tool to probe contact forces and impulses for wider range of problems. Finally, We examined the parallel potential of impulse-based method using domain decomposition strategy, which showed that the current code was sped-up with few processors, but using more processors will stall or even deteriorate scalability due to severe workload imbalance and

unaffordable communication overheads. This aspect of problem deserves further evaluation of different strategies and data abstraction in future work.

Chapter 6

Conclusions and Future Work

The primary objective of the research presented herein was to use the results of detailed X-Ray Computed Tomography (XRCT) characterization of the fabric of a naturally deposited sand in order to build a high fidelity micromechanical DEM model and then explore the effect of soil fabric on the macroscopic, mechanical behavior of the sand. Samples of fine sand obtained from a shoal in the San Francisco Bay scanned and tested in miniature triaxial tests by Garcia et al. (2022) provided the data used for the model development.

The processing of the XRCT data on the fine sand used in this study presented a significant challenge because of the intimate packing of the grains and the presence of different minerals, including fine clay adhesions. Thus, in order to obtain high fidelity representation of the sand fabric we explored different techniques to improve the grain segmentation process. As a first step we used non-local means filter to denoise the images. This method preserves edge sharpness and significantly reduces noise. Then a learning-based image super-resolution method was utilized to enhance the resolution of the images based on sparse signal representation. We found that this method outperforms other methods of image super-resolution in our application. Finally, we performed the image segmentation using HMRF with WEM algorithm in which each group of pixels (void, water, solid etc.) is modelled as a distinct Gaussian and the spatial connectivity is imposed via a stochastic Markov network. We found that this approach increased the accuracy and robustness of the image reconstruction and much improved binary segmentation of XRCT images compared to to conventional methods such as Otsu's method or K-means. As a final step, the avatars for use in DEM were reconstructed from segmented images using a DRLSE algorithm. This approach produced best fidelity, digital reconstruction of the sand fabric.

The principal challenge in being able to perform numerical experiments using micromechanical model of the sand was having an efficient DEM code. To this end a new parallel LS-DEM code was developed based on an existing framework. The new code uses the binning algorithm to reduce the computational complexity from $O(n^2)$ to $O(n)$. The code maps relationship between bins and grains with linked-list like data structure and considers MPI communication in two major parts: border/halo exchange and across-block migration. The resulted numerical experiments show that the code has an excellent weak scalability numer-

ically and has the potential for simulating large scale DEM problems with complex-shaped grains. An important advantage of the MPI implementation is that the code is able to run on wide variety of parallel systems, including shared-memory computers and clustered systems, which makes the code highly portable. Additionally, a quick, efficient code enables systematic parameter exploration and full-scale simulation in other applications.

The new LS-DEM code was then used to replicate the triaxial compression tests performed on undisturbed sand samples and to demonstrate the feasibility of modeling complex natural fabric of sands. Both micro- and macromechanical behaviors of natural materials were well-captured and validated with experimental data, which includes the initial stiffness, peak mobilized friction angle, force chain formation, shear band pattern and fabric evolution. A The results of the numerical modeling show that the primary source of peak strength of sand is the mechanical interlocking between irregularly shaped grains; thus, even the simplest contact model is capable of reproducing both micro- and macro-mechanical responses consistent with experimental data, provided that the microstructure of the grain is captured with sufficient fidelity using high quality scans. Flexible membrane simulations with a rotatable loading platen were found to better predict not only stress-strain and volumetric response, but also the onset and growth of strain localization, allowing them to accurately match experimentally observed relationships between deviatoric stress and mobilized friction angle with axial shortening for uncemented sample.

Finally, we have investigated the viability of modeling dynamic problems with newly formulated impulse-based LS-DEM. The new formulation has both numerical attractions and challenges. While it is stable, fast and energy conservative, it may be numerically stiff when the assembly has a substantial mass difference or badly reconstructed particles as a result of poor image resolution. We demonstrated the feasibility of modeling deformable structures in the rigid body framework and proposed several enhancements to improve the convergence of collision resolution, including a hybrid time integration scheme to separately handle at rest contact and dynamic collision. We then demonstrated that LS-DEM is a generalizable framework which can accommodate any arbitrarily shaped topography and take algorithmic advantages for particle-topography interaction resolution. Specifically, the new formulation allows efficient modeling of granular flows such as rock avalanches with realistic geometry representation without having to assume equivalent fluid behavior as has been done in the past.

The new, parallel implementation of LS-DEM has potential for further enhancements for other applications. The most immediate may to add the capability for modeling particle fracture and disintegration in shear which frequently occurs in granular materials. Modeling heat generation in high velocity simulation of frictional materials is another potential enhancement for exploring phenomena such as rock avalanches. The parallel implementation of the impulse-based LS-DEM deserves additional exploration because the presence of contact islands fundamentally undermines the concept of domain decomposition and hence necessitates a new approach to domain partitioning and communication between processors. Probably the most significant future enhancement would be to integrate liquid phase into the LS-DEM framework and examine the phase front using LS curve evolution techniques.

Thus, the effect of capillary pressure drawing irregularly shaped particles together can be studied at the micro level. Modeling the third phase may be difficult since it requires the underlying mesh to keep track of the deformation of liquid and air phase. As both LS based front propagation and multi-phase flow are often evolved on Cartesian grids, this fact leads to the continual update of mesh surrounding the solid phase. The problem might be resolved with the immersed boundary method, which incorporates the solid-fluid interaction as an additional force component in the Navier-Stokes equation and imposes slip and velocity compatibility as boundary conditions.

Finally, the methodologies developed in this thesis are not tied to granular materials in one particular stress path but also open a door to simulate other families of materials, which can fracture, deform, or flow under more complex loading conditions. The parallel LS-DEM is promising in that it can capture object morphology at highest level and is backward compatible, allowing researchers to choose the appropriate geometry precision, model scale, and dynamic type for the problem at hand.

Bibliography

Abed Zadeh, A., Barés, J., Brzinski, T. A., Daniels, K. E., Dijksman, J., Docquier, N., Everitt, H. O., Kollmer, J. E., Lantsoght, O., Wang, D., et al. (2019). Enlightening force chains: A review of photoelasticimetry in granular matter. *Granular Matter*, *21*(4), 1–12.

Abriak, N., & Caron, J.-F. (2006). Experimental study of shear in granular media. *Advanced Powder Technology*, *17*(3), 297–318.

Amirrahmat, S., Druckrey, A. M., Alshibli, K. A., & Al-Raoush, R. I. (2019). Micro shear bands: Precursor for strain localization in sheared granular materials. *Journal of Geotechnical and Geoenvironmental Engineering*, *145*(2), 04018104.

Amritkar, A., Deb, S., & Tafti, D. (2014). Efficient parallel cfd-dem simulations using openmp. *Journal of Computational Physics*, *256*, 501–519.

Antony, S. J. (2000). Evolution of force distribution in three-dimensional granular media. *Physical Review E*, *63*(1), 011302.

Asai, M., Li, Y., Chandra, B., & Takase, S. (2021). Fluid–rigid-body interaction simulations and validations using a coupled stabilized isph–dem incorporated with the energy-tracking impulse method for multiple-body contacts. *Computer Methods in Applied Mechanics and Engineering*, *377*, 113681.

Aubert, G., Kornprobst, P., & Aubert, G. (2006). *Mathematical problems in image processing: Partial differential equations and the calculus of variations* (Vol. 147). Springer.

Bandeira, A. A., & Zohdi, T. I. (2019). 3d numerical simulations of granular materials using dem models considering rolling phenomena. *Computational Particle Mechanics*, *6*(1), 97–131.

Baraff, D. (1989). Analytical methods for dynamic simulation of non-penetrating rigid bodies. *Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, 223–232.

Barner, K. E., Arce, G. R., & Lin, J. (1992). On the performance of stack filters and vector detection in image restoration. *Circuits, Systems and Signal Processing*, *11*(1), 153–169.

Barzel, R., & Barr, A. H. (1988). A modeling system based on dynamic constraints. *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, 179–188.

Baugh Jr, J. W., & Konduri, R. (2001). Discrete element modelling on a cluster of workstations. *Engineering with Computers*, *17*(1), 1–15.

Bender, J. (2007). Impulse-based dynamic simulation in linear time. *Computer Animation and Virtual Worlds*, *18*(4-5), 225–233.

Carrato, S., Ramponi, G., & Marsi, S. (1996). A simple edge-sensitive image interpolation filter. *Proceedings of 3rd IEEE international conference on image processing*, *3*, 711–714.

Caselles, V., Kimmel, R., & Sapiro, G. (1997). Geodesic active contours. *International journal of computer vision*, *22*(1), 61–79.

Chang, B., & Colgate, J. E. (1997). Real-time impulse-based simulation of rigid body systems for haptic display. *Proceedings of the 1997 ASME international mechanical engineering congress and exhibition*, 1–8.

Chang, H., Yeung, D.-Y., & Xiong, Y. (2004). Super-resolution through neighbor embedding. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, *1*, I–I.

Cho, N. a., Martin, C., & Sego, D. (2007). A clumped particle model for rock. *International journal of rock mechanics and mining sciences*, *44*(7), 997–1010.

Chorley, M. J., & Walker, D. W. (2010). Performance analysis of a hybrid mpi/openmp application on multi-core clusters. *Journal of Computational Science*, *1*(3), 168–174.

Christoffersen, J., Mehrabadi, M. M., & Nemat-Nasser, S. (1981). A micromechanical description of granular material behavior.

Cui, L., & O'Sullivan, C. (2005). Development of a mixed boundary environment for axisymmetric dem analyses. *Proc. 5th Int. Conf. on Micromechanics of Granular Media, Stuttgart*, *1*, 301–305.

Cundall, P. A., & Strack, O. D. (1979). A discrete numerical model for granular assemblies. *geotechnique*, *29*(1), 47–65.

Cuthill, E., & McKee, J. (1969). Reducing the bandwidth of sparse symmetric matrices. *Proceedings of the 1969 24th national conference*, 157–172.

Da Cruz, F., Emam, S., Prochnow, M., Roux, J.-N., & Chevoir, F. (2005). Rheophysics of dense granular materials: Discrete simulation of plane shear flows. *Physical Review E*, *72*(2), 021309.

Daniels, K. E., Kollmer, J. E., & Puckett, J. G. (2017). Photoelastic force measurements in granular materials. *Review of Scientific Instruments*, *88*(5), 051808.

De Bono, J., McDowell, G., & Wanatowski, D. (2012). Discrete element modelling of a flexible membrane for triaxial testing of granular material at high pressures. *Géotechnique Letters*, *2*(4), 199–203.

Donoho, D. L. (2006). For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, *59*(6), 797–829.

Evans, D. J., & Murad, S. (1977). Singularity free algorithm for molecular dynamics simulation of rigid polyatomics. *Molecular physics*, *34*(2), 327–331.

Fakhimi, A., & Villegas, T. (2007). Application of dimensional analysis in calibration of a discrete element model for rock deformation and fracture. *Rock Mechanics and Rock Engineering*, *40*(2), 193–211.

Fu, R., Hu, X., & Zhou, B. (2017). Discrete element modeling of crushable sands considering realistic particle shape effect. *Computers and Geotechnics*, *91*, 179–191.

Garboczi, E. J. (2002). Three-dimensional mathematical analysis of particle shape using x-ray tomography and spherical harmonics: Application to aggregates used in concrete. *Cement and concrete research*, *32*(10), 1621–1638.

Garcia, E., Ando, E., Viggiani, G., & Sitar, N. (2022). Influence of depositional fabric on mechanical properties of naturally deposited sands (in press). *Geotechnique*.

Garcia, X., Latham, J.-P., XIANG, J.-s., & Harrison, J. (2009). A clustered overlapping sphere algorithm to represent real particles in discrete element modelling. *Geotechnique*, *59*(9), 779–784.

Gerig, G., Kikinis, R., Kuoni, W., Von Schulthess, G. K., & Kübler, O. (1992). Semiautomated roi analysis in dynamic mri studies: Part i: Image analysis tools for automatic correction of organ displacements. *IEEE Trans. Image Processing*, *11*(2), 221–232.

Gopalakrishnan, P., & Tafti, D. (2013). Development of parallel dem for the open source code mfix. *Powder technology*, *235*, 33–41.

Gray, A., & Moore, A. (2000). N-body'problems in statistical learning. *Advances in neural information processing systems*, *13*.

Grest, G. S., Dünweg, B., & Kremer, K. (1989). Vectorized link cell fortran code for molecular dynamics simulations for a large number of particles. *Computer Physics Communications*, *55*(3), 269–285.

Guendelman, E., Bridson, R., & Fedkiw, R. (2003). Nonconvex rigid bodies with stacking. *ACM transactions on graphics (TOG)*, *22*(3), 871–878.

Guillemaud, R., & Brady, M. (1997). Estimating the bias field of mr images. *IEEE Transactions on Medical imaging*, *16*(3), 238–251.

Guo, P. (2012). Critical length of force chains and shear band thickness in dense granular materials. *Acta Geotechnica*, *7*(1), 41–55.

Haier, E., Lubich, C., & Wanner, G. (2006). *Geometric numerical integration: Structure-preserving algorithms for ordinary differential equations*. Springer.

Hazzar, L., Nuth, M., & Chekired, M. (2020). Dem simulation of drained triaxial tests for glass-beads. *Powder Technology*, *364*, 123–134.

Hazzard, J. F., Collins, D. S., Pettitt, W. S., & Young, R. P. (2002). Simulation of unstable fault slip in granite using a bonded-particle model. *The mechanism of induced seismicity* (pp. 221–245). Springer.

Henty, D. S. (2000). Performance of hybrid message-passing and shared-memory parallelism for discrete element modeling. *SC'00: Proceedings of the 2000 ACM/IEEE Conference on Supercomputing*, 10–10.

Ho, K. K., Koo, R. C., & Kwan, J. S. (2021). Mitigation of debris flows—research and practice in hong kong. *Environmental & Engineering Geoscience*, *27*(2), 231–243.

Hou, H., & Andrews, H. (1978). Cubic splines for image interpolation and digital filtering. *IEEE Transactions on acoustics, speech, and signal processing*, *26*(6), 508–517.

Huisken, G. (1984). Flow by mean curvature of convex surfaces into spheres. *Journal of Differential Geometry*, *20*(1), 237–266.

Ishihara, K. (1993). Liquefaction and flow failure during earthquakes. *Geotechnique*, *43*(3), 351–451.

Itasca, C. (2004). Pfc2d (particle flow code in two dimensions). *Minneapolis, Minnesota, USA*.

Itasca, P. (1998). 2.00 particle flow code in two dimensions. *Minneapolis Minnesota*.

Jagadish, H. V., Ooi, B. C., Tan, K.-L., Yu, C., & Zhang, R. (2005). Idistance: An adaptive b+-tree based indexing method for nearest neighbor search. *ACM Transactions on Database Systems (TODS)*, *30*(2), 364–397.

Kačianauskas, R., Maknickas, A., Kačeniauskas, A., Markauskas, D., & Balevičius, R. (2010). Parallel discrete element simulation of poly-dispersed granular material. *Advances in Engineering Software*, *41*(1), 52–63.

Karp, A. H., & Flatt, H. P. (1990). Measuring parallel processor performance. *Communications of the ACM*, *33*(5), 539–543.

Kawamoto, R., Andò, E., Viggiani, G., & Andrade, J. E. (2016). Level set discrete element method for three-dimensional computations with triaxial case study. *Journal of the Mechanics and Physics of Solids*, *91*, 1–13.

Kawamoto, R., Andò, E., Viggiani, G., & Andrade, J. E. (2018). All you need is shape: Predicting shear banding in sand with ls-dem. *Journal of the Mechanics and Physics of Solids*, *111*, 375–392.

Kawamoto, R. Y. (2018). *The avatar paradigm in granular materials* (Doctoral dissertation). California Institute of Technology.

Kim, H., Han, J., & Han, T. Y.-J. (2020). Machine vision-driven automatic recognition of particle size and morphology in sem images. *Nanoscale*, *12*(37), 19461–19469.

Kim, K. I., & Kwon, Y. (2010). Single-image super-resolution using sparse regression and natural image prior. *IEEE transactions on pattern analysis and machine intelligence*, *32*(6), 1127–1133.

Kuhn, M. R. (1999). Structured deformation in granular materials. *Mechanics of materials*, *31*(6), 407–429.

Lai, Z., & Chen, Q. (2019). Reconstructing granular particles from x-ray computed tomography using the tws machine learning tool and the level set method. *Acta Geotechnica*, *14*(1), 1–18.

Lam, W.-K., & Tatsuoka, F. (1988). Triaxial compressive and extension strength of sand affected by strength anisotropy and sample slenderness. *Advanced triaxial testing of soil and rock*. ASTM International.

Lee, H., Battle, A., Raina, R., & Ng, A. (2007). In schölkopfb. plattj. hoffmant.(eds.), efficient sparse coding algorithms (vol. 19, pp. 801–808). *Advances in Neural Information Processing Systems*.

Lee, S. J., & Hashash, Y. M. (2015). Idem: An impulse-based discrete element method for fast granular dynamics. *International Journal for Numerical Methods in Engineering*, *104*(2), 79–103.

Lee, S. J., Hashash, Y. M., & Nezami, E. G. (2012). Simulation of triaxial compression tests with polyhedral discrete elements. *Computers and Geotechnics*, *43*, 92–100.

Li, C., Xu, C., Gui, C., & Fox, M. D. (2005). Level set evolution without re-initialization: A new variational formulation. *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, *1*, 430–436.

Li, C., Xu, C., Gui, C., & Fox, M. D. (2010). Distance regularized level set evolution and its application to image segmentation. *IEEE transactions on image processing*, *19*(12), 3243–3254.

Li, Y., Asai, M., Chandra, B., & Isshiki, M. (2021). Energy-tracking impulse method for particle-discretized rigid-body simulations with frictional contact. *Computational Particle Mechanics*, *8*(2), 237–258.

Liang, W., & Zhao, J. (2019). Multiscale modeling of large deformation in geomechanics. *International Journal for Numerical and Analytical Methods in Geomechanics*, *43*(5), 1080–1114.

Lim, K.-W., & Andrade, J. E. (2014). Granular element method for three-dimensional discrete element calculations. *International Journal for Numerical and Analytical Methods in Geomechanics*, *38*(2), 167–188.

Lim, K.-W., Krabbenhoft, K., & Andrade, J. E. (2014). A contact dynamics approach to the granular element method. *Computer Methods in Applied Mechanics and Engineering*, *268*, 557–573.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, *1*(14), 281–297.

Maknickas, A., Kačeniauskas, A., Kačianauskas, R., Balevičius, R., & Džiugys, A. (2006). Parallel dem software for simulation of granular media. *Informatica*, *17*(2), 207–224.

MiDi, G. (2004). On dense granular flows. *The European Physical Journal E*, *14*, 341–365.

Mirtich, B., & Canny, J. (1995). Impulse-based simulation of rigid bodies. *Proceedings of the 1995 symposium on Interactive 3D graphics*, 181–ff.

Mirtich, B. V. (1996). *Impulse-based dynamic simulation of rigid body systems*. University of California, Berkeley.

Mitchell, J. K., & Soga, K. (2005). *Fundamentals of soil behavior* (Vol. 3). John Wiley & Sons New York.

Mollon, G., Quacquarelli, A., Andò, E., & Viggiani, G. (2020). Can friction replace roughness in the numerical simulation of granular materials? *Granular Matter*, *22*(2), 1–16.

Mollon, G., & Zhao, J. (2012). Fourier–voronoi-based generation of realistic samples for discrete modelling of granular materials. *Granular matter*, *14*(5), 621–638.

Mondain-Monval, O., Espert, A., Omarjee, P., Bibette, J., Leal-Calderon, F., Philip, J., & Joanny, J.-F. (1998). Polymer-induced repulsive forces: Exponential scaling. *Physical review letters*, *80*(8), 1778.

Moore, M., & Wilhelms, J. (1988). Collision detection and response for computer animation. *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, 289–298.

Muja, M., & Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)*, *2*(331-340), 2.

Mulilis, J. P., Seed, H. B., Chan, C. K., Mitchell, J. K., & Arulanandan, K. (1977). Effects of sample preparation on sand liquefaction. *Journal of the Geotechnical Engineering Division*, *103*(2), 91–108.

Munjiza, A., & Andrews, K. (1998). Nbs contact detection algorithm for bodies of similar size. *International Journal for Numerical Methods in Engineering*, *43*(1), 131–149.

Ng, T.-T. (2006). Input parameters of discrete element methods. *Journal of Engineering Mechanics*, *132*(7), 723–729.

Nouguier-Lehon, C., Cambou, B., & Vincens, E. (2003). Influence of particle shape and angularity on the behaviour of granular materials: A numerical analysis. *International journal for numerical and analytical methods in geomechanics*, *27*(14), 1207–1226.

Ochiai, H., & Lade, P. V. (1983). Three-dimensional behavior of sand with anisotropic fabric. *Journal of Geotechnical Engineering*, *109*(10), 1313–1328.

Oda, M. (1972). The mechanism of fabric changes during compressional deformation of sand. *Soils and foundations*, *12*(2), 1–18.

Oda, M., Nemat-Nasser, S., & Konishi, J. (1985). Stress-induced anisotropy in granular masses. *Soils and foundations*, *25*(3), 85–97.

Osher, S., & Sethian, J. A. (1988). Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *Journal of computational physics*, *79*(1), 12–49.

O'Sullivan, C., & Bray, J. D. (2004). Selecting a suitable time step for discrete element simulations that use the central difference time integration scheme. *Engineering Computations*.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, *9*(1), 62–66.

Owen, D., & Feng, Y. (2001). Parallelised finite/discrete element simulation of multi-fracturing solids and discrete systems. *Engineering computations*.

Park, J.-W., & Song, J.-J. (2009). Numerical simulation of a direct shear test on a rock joint using a bonded-particle model. *International Journal of Rock Mechanics and Mining Sciences*, *46*(8), 1315–1328.

Penzien, J. (1993). *Dynamics of structures*. McGraw-Hill.

Perona, P., & Malik, J. (1990). Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, *12*(7), 629–639.

Peters, J. F., Hopkins, M. A., Kala, R., & Wahl, R. E. (2009). A poly-ellipsoid particle for non-spherical discrete element method. *Engineering Computations*.

Pickup, L., Roberts, S. J., & Zisserman, A. (2003). A sampled texture prior for image super-resolution. *Advances in neural information processing systems*, *16*.

Potyondy, D. O., & Cundall, P. (2004). A bonded-particle model for rock. *International journal of rock mechanics and mining sciences*, *41*(8), 1329–1364.

Radjai, F., Jean, M., Moreau, J.-J., & Roux, S. (1996). Force distributions in dense two-dimensional granular systems. *Physical review letters*, *77*(2), 274.

Rice, J. (1976). Theoretical and applied mechanics. *Proc. of the 14th IUTAM Congress, North-Holland, Amsterdam, Netherlands, 1976*, 207–220.

Rodgers, D. P. (1985). Improvements in multiprocessor system design. *ACM SIGARCH Computer Architecture News*, *13*(3), 225–231.

Rorato, R., Arroyo, M., Gens, A., Andò, E., & Viggiani, G. (2021). Image-based calibration of rolling resistance in discrete element models of sand. *Computers and Geotechnics*, *131*, 103929.

Roscoe, K. H. (1970). The influence of strains in soil mechanics. *Geotechnique*, *20*(2), 129–170.

Rudin, L. I., Osher, S., & Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, *60*(1-4), 259–268.

Rudnicki, J. W., & Rice, J. (1975). Conditions for the localization of deformation in pressure-sensitive dilatant materials. *Journal of the Mechanics and Physics of Solids*, *23*(6), 371–394.

Salgado, R., Bandini, P., & Karim, A. (2000). Shear strength and stiffness of silty sand. *Journal of geotechnical and geoenvironmental engineering*, *126*(5), 451–462.

Satake, M. (1982). Fabric tensor in granular materials. *IUTAM Conference on Deformation and Flow of Granular Materials, 1982*, 63–68.

Schanz, T., & Vermeer, P. (1996). Angles of friction and dilatancy of sand. *Geotechnique*, *46*(1), 145–151.

Sethian, J. A. (1996). A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, *93*(4), 1591–1595.

Sethian, J. A. (1999). Fast marching methods. *SIAM review*, *41*(2), 199–235.

Setiasabda, E. Y. (2020). *Material point method for large deformation modeling in geomechanics using isoparametric elements*. University of California, Berkeley.

Stronge, W. J. (2018). *Impact mechanics*. Cambridge university press.

Sun, J., Xu, Z., & Shum, H.-Y. (2008). Image super-resolution using gradient profile prior. *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 1–8.

Sun, J., Zheng, N.-N., Tao, H., & Shum, H.-Y. (2003). Image hallucination with primal sketch priors. *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, *2*, II–729.

Tamadondar, M. R., de Martın, L., & Rasmuson, A. (2019). Agglomerate breakage and adhesion upon impact with complex-shaped particles. *AIChE Journal*, *65*(6), e16581.

Tang, X., Paluszny, A., & Zimmerman, R. W. (2014). An impulse-based energy tracking method for collision resolution. *Computer Methods in Applied Mechanics and Engineering*, *278*, 160–185.

Taylor, M. A., Garboczi, E., Erdogan, S., & Fowler, D. (2006). Some properties of irregular 3-d particles. *Powder Technology*, *162*(1), 1–15.

Terzaghi, K. (1955). Influence of geological factors on the engineering properties of sediments.

Thornton, C. (2000). Numerical simulations of deviatoric shear deformation of granular media. *Géotechnique*, *50*(1), 43–53.

Tomasi, C., & Manduchi, R. (1998). Bilateral filtering for gray and color images. *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*, 839–846.

Vaid, Y. P., Sivathayalan, S., & Stedman, D. (1999). Influence of specimen-reconstituting method on the undrained response of sand. *Geotechnical Testing Journal*, *22*(3), 187–195.

Vlahinić, I., Andrade, J., Andö, E., & Viggiani, G. (2013). From 3d tomography to physics-based mechanics of geomaterials. *Computing in civil engineering (2013)* (pp. 339–345).

Vlahinić, I., Andò, E., Viggiani, G., & Andrade, J. E. (2014). Towards a more accurate characterization of granular media: Extracting quantitative descriptors from tomographic images. *Granular Matter*, *16*(1), 9–21.

Walther, J. H., & Sbalzarini, I. F. (2009). Large-scale parallel discrete element simulations of granular flow. *Engineering Computations*.

Walton, O., & Braun, R. (1993). *Simulation of rotary-drum and repose tests for frictional spheres and rigid sphere clusters* (tech. rep.). Lawrence Livermore National Lab., CA (United States).

Wang, X., Nie, Z., Gong, J., & Liang, Z. (2021). Random generation of convex aggregates for dem study of particle shape effect. *Construction and Building Materials*, *268*, 121468.

Washington, D. W., & Meegoda, J. N. (2003). Micro-mechanical simulation of geotechnical problems using massively parallel computers. *International journal for numerical and analytical methods in geomechanics*, *27*(14), 1227–1234.

Wells, W. M., Grimson, W. E. L., Kikinis, R., & Jolesz, F. A. (1996). Adaptive segmentation of mri data. *IEEE transactions on medical imaging*, *15*(4), 429–442.

Wiebicke, M., Andò, E., Viggiani, G., & Herle, I. (2020). Measuring the evolution of contact fabric in shear bands with x-ray tomography. *Acta Geotechnica*, *15*(1), 79–93.

Williams, J. R., Perkins, E., & Cook, B. (2004). A contact algorithm for partitioning n arbitrary sized objects. *Engineering Computations*.

Wood, D., & Maeda, K. (2008). Changing grading of soil: Effect on critical states. *Acta Geotechnica*, *3*(1), 3–14.

Wriggers, P., & Laursen, T. A. (2006). *Computational contact mechanics* (Vol. 2). Springer.

Wright, J., Yang, A. Y., Ganesh, A., Sastry, S. S., & Ma, Y. (2008). Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, *31*(2), 210–227.

Wu, M., Wang, J., Russell, A., & Cheng, Z. (2021). Dem modelling of mini-triaxial test based on one-to-one mapping of sand particles. *Géotechnique*, *71*(8), 714–727.

Yan, B., & Regueiro, R. (2018a). Comparison between o (n2) and o (n) neighbor search algorithm and its influence on superlinear speedup in parallel discrete element method (dem) for complex-shaped particles. *Engineering Computations*.

Yan, B., & Regueiro, R. A. (2018b). A comprehensive study of mpi parallelism in three-dimensional discrete element method (dem) simulation of complex-shaped granular particles. *Computational Particle Mechanics*, *5*(4), 553–577.

Yan, B., & Regueiro, R. A. (2019). Comparison between pure mpi and hybrid mpi-openmp parallelism for discrete element method (dem) of ellipsoidal and poly-ellipsoidal particles. *Computational Particle Mechanics*, *6*(2), 271–295.

Yang, J., Wright, J., Huang, T. S., & Ma, Y. (2010). Image super-resolution via sparse representation. *IEEE transactions on image processing*, *19*(11), 2861–2873.

Yoshimine, M., Ishihara, K., & Vargas, W. (1998). Effects of principal stress direction and intermediate principal stress on undrained shear behavior of sand. *Soils and Foundations*, *38*(3), 179–188.

Zhang, Y., Brady, M., & Smith, S. (2001). Segmentation of brain mr images through a hidden markov random field model and the expectation-maximization algorithm. *IEEE transactions on medical imaging*, *20*(1), 45–57.

Zhao, D., Nezami, E. G., Hashash, Y. M., & Ghaboussi, J. (2006). Three-dimensional discrete element simulation for granular materials. *Engineering Computations*.

Zhao, S., & Zhao, J. (2019). A poly-superellipsoid-based approach on particle morphology for dem modeling of granular media. *International Journal for Numerical and Analytical Methods in Geomechanics*, *43*(13), 2147–2169.

Zhao, S., & Zhao, J. (2021). Sudodem: Unleashing the predictive power of the discrete element method on simulation for non-spherical granular particles. *Computer Physics Communications*, *259*, 107670.

Zheng, J., An, X., & Huang, M. (2012). Gpu-based parallel algorithm for particle contact detection and its application in self-compacting concrete flow simulations. *Computers & Structures*, *112*, 193–204.

Zhou, B., Wang, J., & Zhao, B. (2015). Micromorphology characterization and reconstruction of sand particles using micro x-ray tomography and spherical harmonics. *Engineering geology*, *184*, 126–137.

Zlatovic, S., & Ishihara, K. (1997). Normalized behavior of very loose non-plastic soils: Effects of fabric. *Soils and foundations*, *37*(4), 47–56.

Zohdi, T., & Wriggers, P. (1999). A domain decomposition method for bodies with heterogeneous microstructure basedon material regularization. *International Journal of Solids and Structures*, *36*(17), 2507–2525.

Zohdi, T. (2007). Particle collision and adhesion under the influence of near-fields. *Journal of Mechanics of Materials and Structures*, *2*(6), 1011–1018.

Zohdi, T. I. (2017). *Modeling and simulation of functionalized materials for additive manufacturing and 3d printing: Continuous and discrete media: Continuum and discrete element methods* (Vol. 60). Springer.

Zohdi, T. I., & Wriggers, P. (2001). Computational micro-macro material testing. *Archives of Computational Methods in Engineering*, *8*(2), 131–228.

Zohdi, T. I., & Wriggers, P. (2004). *An introduction to computational micromechanics* (Vol. 20). Springer Science & Business Media.

Zohdi, T. (2004a). A computational framework for agglomeration in thermochemically reacting granular flows. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, *460*(2052), 3421–3445.

Zohdi, T. (2004b). Staggering error control of a class of inelastic processes in random microheterogeneous solids. *International journal of non-linear mechanics*, *39*(2), 281–297.

Zohdi, T. (2012). Estimation of electrical heating load-shares for sintering of powder mixtures. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *468*(2144), 2174–2190.

Zohdi, T. (2014). Impact and penetration resistance of network models of coated lightweight fabric shielding. *GAMM-Mitteilungen*, *37*(1), 124–150.

Zohdi, T. (2003). Genetic design of solids possessing a random–particulate microstructure. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, *361*(1806), 1021–1043.

Zohdi, T. (2010). Simulation of coupled microscale multiphysical-fields in particulate-doped dielectrics with staggered adaptive fdtd. *Computer Methods in Applied Mechanics and Engineering*, *199*(49-52), 3250–3269.

Zohdi, T. (2013). Numerical simulation of charged particulate cluster-droplet impact on electrified surfaces. *J Comput Phys*, *233*, 509–526.