UNIVERSIDADE FEDERAL DE ALAGOAS

CENTRO DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA CIVIL

LUCAS GOUVEIA OMENA LOPES

PARTICLE PACKING STRATEGIES FOR MODELLING GRANULAR MEDIA

Orientador: Prof. Dr. William Wagner Matos Lira
Coorientador: Dr. Diogo Tenório Cintra

MACEIÓ-AL
2019

LUCAS GOUVEIA OMENA LOPES

PARTICLE PACKING STRATEGIES FOR MODELLING GRANULAR MEDIA

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Civil da Universidade Federal de Alagoas, como requisito para obtenção do grau de Mestre em Estruturas.
Orientador: Prof. Dr. William Wagner Matos Lira
Coorientador: Dr. Diogo Tenório Cintra

MACEIÓ-AL

2019

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Bibliotecário: Marcelino de Carvalho

L864p Lopes, Lucas Gouveia Omena.
Particle packing strategies for modelling granular media / Lucas Gouveia Omena Lopes. - 2019.
82 f. : il.

Orientador: William Wagner Matos Lira.
Coorientador: Diogo Tenório Cintra.
Dissertação (Mestrado em Engenharia Civil) – Universidade Federal de Alagoas. Centro de Tecnologia. Maceió, 2019.

Texto em inglês.
Bibliografia: f. 80-82.

1. Modelagem de meios granulares. 2. Geração de partículas. 3. Método dos elementos discretos. 4. Otimização. 5. Computação de alto desempenho. I. Título.

CDU:621.7-492.3

Universidade Federal de Alagoas – UFAL
Unidade Acadêmica Centro de Tecnologia – CTEC
Programa de Pós-Graduação em Engenharia Civil – PPGEC

PARTICLE PACKING STRATEGIES FOR MODELLING GRANULAR MEDIA

LUCAS GOUVEIA OMENA LOPES

Dissertação submetida à banca examinadora do Programa de Pós-Graduação em Engenharia Civil da Universidade Federal de Alagoas e aprovada no dia 03 do mês de outubro do ano de 2019.

Banca Examinadora:

Prof. Dr. William Wagner Matos Lira
(Orientador – PPGEC/UFAL)

Pesquisador Dr. Diogo Tenório Cintra
(Coorientador - LCCV/UFAL)

Pesquisador Dr. Hélvio de Farias Costa Peixoto
(Avaliador Externo– LCCV/UFAL)

Prof. Dr. Thales Miranda de Almeida Vieira
(Avaliador Externo – IM/IC/UFAL)

Campus A. C. Simões, Av. Lourival de Melo Mota, S/N
Tabuleiro do Martins – CEP 57072-970 – Maceió – Alagoas
Tel/Fax: (82) 3214-1863
E-mail: ppgec@ctec.ufal.br
Homepage: www.ctec.ufal.br/posgraduacao/ppgec

RESUMO

LOPES, Lucas Gouveia Omena. ESTRATÉGIAS DE EMPACOTAMENTO DE PARTÍCULAS PARA MODELAGEM DE MEIOS GRANULARES. 2019. 82f. Dissertação – Centro de Tecnologia, Universidade Federal de Alagoas, Maceió-AL, 2019.

Este trabalho apresenta estratégias para modelagem de meios granulares que mimetizam características presentes em agregados particulados reais, sobretudo solos. Busca-se alcançar modelos granulares homogêneos que respeitem taxas de preenchimento e tamanhos de grãos estabelecidos. As estratégias podem ser utilizadas por simuladores de meios particulados na criação do arranjo inicial dos grãos, como aqueles que utilizam Método dos Elementos Discretos (MED). Simulações numéricas de fenômenos de fragmentação, fraturamento, impacto e colisão, que envolvem solos e rochas como meios integrantes de problemas geomecânicos são exemplos de aplicações do MED. A literatura divide as técnicas de geração de meio particulado em duas categorias: procedimentos geométricos e dinâmicos. Neste trabalho são exploradas três metodologias alternativas às existentes na literatura: i) minimização da distâncias; ii) técnica de separação geométrica serial; e iii) técnica de separação geométrica alternativa paralela. As abordagens apresentadas tratam de procedimentos geométricos a partir de um modelo de entrada com partículas em posições aleatórias. As sobreposições entre as partículas é eliminada durante os procedimentos das metodologias propostas, e um modelo de saída é alcançado. Uma importante contribuição deste trabalho é o desenvolvimento de técnicas que melhoram o tempo de processamento para modelagem utilizando uma implementação paralela com foco nas GPUs (*Graphics Processing Units*). Modelos granulares gerados a partir de dados reais do solo são apresentados para validação das estratégias propostas. Os resultados mostram que as técnicas são capazes de gerar modelos granulares com características próximas a de agregados reais. A utilização de estratégias paralelas em GPU também apresenta ganhos significativos, alcançando *speedups* acima de 200.

Palavras-chaves: Modelagem de Meios Granulares. Geração de Partículas. Método dos Elementos Discretos. Otimização. Computação de Alto Desempenho.

ABSTRACT

LOPES, Lucas Gouveia Omena. PARTICLE PACKING STRATEGIES FOR MODELLING GRANULAR MEDIA. 2019. 82f. Dissertation – Centro de Tecnologia, Universidade Federal de Alagoas, Maceió-AL, 2019.

This work presents strategies for modeling numerical granular media that reproduce characteristics present in real particulate aggregates, such as soils. The presented methodologies have the objective to achieve homogeneous granular models respecting prescribed filling rate and grain size distributions. The strategies can be used by simulators of particulate media to create the initial particle arrangement, such as those used in the Discrete Element Method (DEM). Examples of DEM applications are numerical simulations of fragmentation, fracturing, impact and collision phenomena, as well as those involving soils and rocks as integral media of geomechanical problems. The literature divides techniques for generating particulate media into two categories: geometric and dynamic procedures. Three alternative methodologies to those present in the literature are explored in this work: i) distance minimization; ii) serial geometric separation technique; and iii) parallel alternative geometric separation technique. The presented approaches deal with geometric procedures starting from an input model with particles in random positions. The overlapping between the particles is eliminated during the proposed methodology procedures, and an output model is delivered. An important contribution of this work is the development of techniques that improve the processing time for modeling using parallel implementation with a focus on GPUs (Graphics Processing Units). Granular models generated from real soil data are presented for validation of the proposed strategies. The results show that the techniques are capable of generating granular models with characteristics close to those of real aggregates. The use of parallel strategies in GPU also presents significant gains, reaching speedups above 200.

Keywords: Granular Media Modeling. Particle Generation. Discrete Element Method. Optimization. High Performace Computing.

ACKNOWLEDGEMENTS

First of all, to God, present in all moments of success or difficulty. Always illuminating my path and blessing my choices.

To my parents, Artur and Selma, for always placing all love and trust, giving all the necessary structure for my education and development as a person and professional.

To my cousins, uncles, godparents and grandparents, who make of the family a solid base in my life, that even in the distance are made present, always giving me support, affection and motivation.

To the advisors William Wagner and Diogo Cintra, always available and attentive, sharing their knowledge and contributing to obtain the best results.

To all the other teachers who were part of my graduation and post graduation, especially those who fulfilled the role of educator sharing their knowledge in an inspiring way, and who contributed to my professional training.

To all my friends and coworkers of the Laboratory of Scientific Computing and Visualization (LCCV), who helped me during all my academic years.

To all my friends that were present during all the good and bad moments of my academic life. In special: to my best friend Edelson; to the Zero Light Breakers Limited Club (Ark Beth (Bela) and Heidi); to the Boardgames with Mortadella (Helvio, Marilia, Helene, Diogo, Ricardo, Felipe); and all other friends that crossed my path, making a little easier this hard journey.

To FAPEAL for their support and funding in research, and PETROBRAS for the development of projects that have resulted in this work.

Sic Parvis Magna
(Sir Francis Drake)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

MED	*Método dos Elementos Discretos*
DEM	*Discrete Element Method*
HPC	*High Performance Computing*
GPU	*Graphics Processing Unit*
NBS	*Non Binary Search*
PPGEC	*Post-graduate Program in Civil Engineering*
UFAL	*Universidade Federal de Alagoas*
WRO	*Well-Reduced Overlapping*

1 INTRODUCTION

The literature presents several theoretical contributions in the field of particle packing methodologies. According to Recarey et al. (RECAREY et al., 2019), modeling using particle packing methods is an alternative to the traditional continuous approach, which is under rapid development lately in the context of Computational Mechanics. Particle methods are solid strategies for granular media modeling, in particular, the Discrete Element Method (DEM). To achieve increasingly significant results in the study of engineering problems involving granulated media, the DEM has become a powerful tool in the numerical simulation of these studies (CUNDALL; STRACK, 1979). Among the classes of problems that the method has been applied more frequently in recent years, are the phenomena of fragmentation, fracturing, impact and collision, in addition to those directly related to soil modeling as an integral part of geomechanical problems (CAMPBELL; CLEARY; HOPKINS, 1995; BROWN et al., 2000; ONATE; ROJEK, 2004; CHANG; TABOADA, 2009).

Recarey el al. (RECAREY et al., 2019) says that similar to the Finite Element Method (FEM), where the media have to be discretized into finite elements, an initial set of particles or " packing " is required in DEM. In a large number of problems, this discretization must meet some important requirements. For example, in geomechanical problems, the representativeness of the soil in the simulation requires that the grain size distribution and filling rate previously specified meet the initial conditions of that soil. Also, it is important to ensure the homogeneity of the particulate medium and the balance of the system. Since obtaining such an initial set is not a trivial task, it is necessary to develop particle filling formulations algorithms, which are essential to perform DEM applications. Despite the diversity of particles used in DEM, spheres, and disks are more often used due to their simplicity. A large number of strategies have been presented for granular modeling with well-known forms, mainly discs, and spheres, for two-dimensional and three-dimensional cases, respectively. These strategies can be subdivided into two distinct groups: dynamic algorithms and geometric algorithms.

Dynamic algorithms are those that use the solution of a motion equation as a mechanism to define the positions of particles. According to Recary et al. (RECAREY et al., 2019) Dynamic techniques include: 1- Isotropic compression, 2- particle growth, 3- gravitational deposition and 4- multilayer compaction. These algorithms use physical representations similar or identical to those used in the DEM solution methodology (LIU; ZHANG; YU, 1999; SIIRIA; YLIRUUSI, 2007). Particulate domains with arbitrary geometries can be generated through dynamic algorithms, whether or not they involve particles of varying sizes. A disadvantage of these algorithms, however, is the computational cost demanded.

The parameters and physical models used in generation often require the calculation of a large number of solution iterations. This is often due to criteria related to the stability of temporal integration algorithms. The limiting factor for the efficiency of these algorithms is not simply the number of particles involved in the generation, but also the number of solution steps required.

Geometric algorithms, also known as Constructive algorithms, are characterized by the use of purely geometric techniques to generate particles on given domain. This feature significantly reduces the processing time required in the generation, since it does not require the solution of equations of motion. Recarey et al. cite that the following construction techniques can be found in the literature: 1- Regular arrangements, 2- Sequential inhibition, and 3- Advance front, with its interior packaging and exterior packaging variants.

Cui and O'Sullivan (CUI; O'SULLIVAN, 2003) presented a geometric technique for particle generation based on an initial mesh of triangular or tetrahedral elements. The technique consists in defining the positions of the particles so that they are inscribed inside the elements of the mesh. A limiting factor for the use of the technique is the need for an initial mesh of elements and the difficulty of controlling the size of the particles.

Another geometric generation technique, presented by Frery et al. (FRERY et al., 2011), consists of positioning particles using the Sequential inhibition based on the theory of point processes, preventing the geometric overlapping of particles from occurring. The particle sizes are initially defined based on representative particle size distribution curves of the soil and the attempt to insert particles into the system until the porosity initially specified is reached.

Using the concept of mesh generation by advancing front, Feng et al. (FENG; HAN; OWEN, 2003) present a particle packing technique where the particle positions are calculated based on information previously included in the domain. The technique is very efficient, but presents some limitations in particular cases of generation of three-dimensional models. To ensure the robustness of the strategy, in this particular case, the complexity of the algorithm grows, therefore compromising its efficiency.

A very simple particle packing technique called compression algorithm is presented by Han et al. (HAN; FENG; OWEN, 2005). In this technique, particles are generated in random positions, with no overlapping between them, and subsequently moved appropriately in a direction of compression.

In the field of geometric algorithms, Labra and Oñate (LABRA; ONATE, 2009) present a particle packing technique of generating dense models. The basic idea of the algorithm is to modify an initial arrangement of particles, repositioning or resizing the particles, through a process of minimizing distances. This procedure uses the Levenberg-Marquardt minimization scheme (LEVENBERG, 1944; MARQUARDT, 1963), also used in the

present work. The proposal of Labra, however, does not guarantee that the particle size distributions and the porosity representative of the studied media are respected.

In summary, dynamic algorithms usually allow more easiness on the particle packing, besides prescribed particle sizes and global force equilibrium. However, the computational cost of these algorithms can make prohibitive their use, especially on large-scale models. Possible solutions to improve the computing time is presented by Baugh and Konduri, and Cintra et al. (BAUGH; KONDURI, 2001; CINTRA et al., 2016a). In their work, they show that it is possible to reduce the convergence time of large DEM simulations using High-Performance Computing (HPC).

The traditional geometric algorithms, however, have difficulty to achieve responses that meet the specified particle sizes and the dynamic equilibrium. However, geometric techniques can be used in certain types of problems, such as fully saturated soils, as described by Frery et al. (FRERY et al., 2011). Frery proposes a method that target to meet both filling ratio and size distribution of real granular models with spatial homogeneity, achieving close results to experimental data of fully saturated soils.

Based on the above explanations, this work proposes methodologies for modeling granular media that reproduces characteristics present in real granular aggregates, especially soils. The presented methodologies has the objective to achieve homogeneous granular models respecting prescribed filling rate and grain size distributions. The strategies can be used by simulators of particulate media in order to create the initial particle arrangement, such as those used in DEM. The strategies presented are: i) distance minimization; ii) geometric separation; and iii) parallel geometric separation. The algorithms do not belong to the classifications described in the literature. An important contribution of this work is the development of techniques that improve the processing time for modeling using parallel implementation with focus in GPUs (Graphics Processing Units). Granular models generated from real soil data are presented for validation of the proposed strategies. All three algorithms show good results meeting the characteristics of real soils. The number of particles and computing time achieved is comparable to state-of-the-art strategies.

1.1 Objective

The objective of this work is the development and validation of three particle packing methodologies for modeling granular media. The proposed strategies seek to achieve homogeneous granular models that respect prescribed filling rate and grain size distribution. The target is to obtain methods with low computational cost, but that produce granular models with output values that meet or are at least close to the specified parameters of real particulate aggregates. The strategies presented are: i) distance minimization; ii) geometric separation; and iii) parallel geometric separation. This work also presents three scientific papers, submitted to reputable journals, that present the studies and methods

developed for each of the proposed techniques.

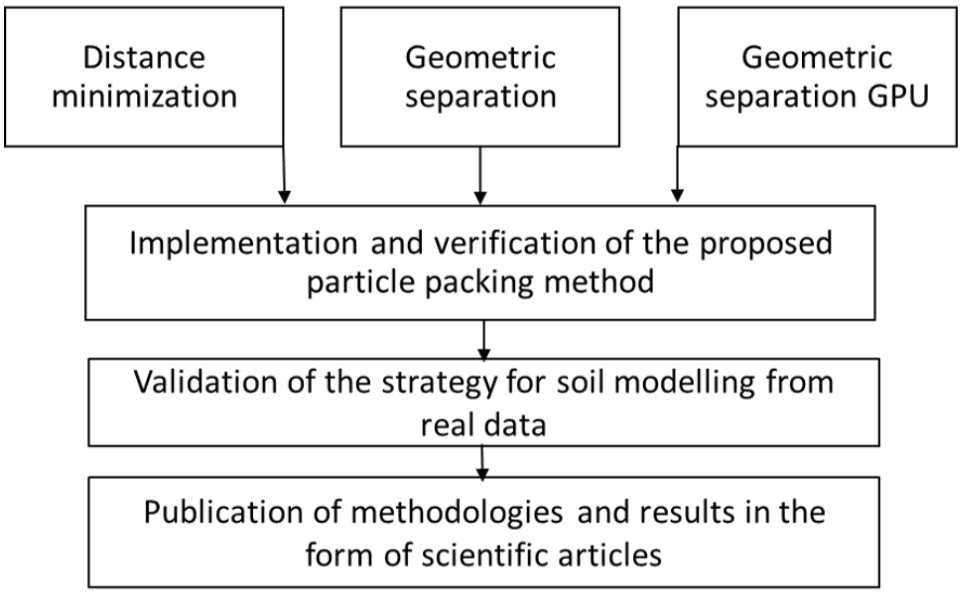
To achieve the main objective, the work presents the following specific objectives:

- Development of a granular media modeling strategy based on the minimization of distances, making use of the Levenberg-Marquadt algorithm;
- Development of a geometric separation strategy for the generation of particulate models;
- Study and Implementation of the parallel version of the geometric separation strategy, making use of HPC with focus in GPUs;
- Application of the strategies in the modeling of real soils, emphasizing the applicability in engineering problems;
- Publication of strategies in the form of scientific articles.

1.2 Methodology

In order to achieve the objectives proposed in this work, the development methodology is divided into three macro stages, as illustrated in Figure 1.

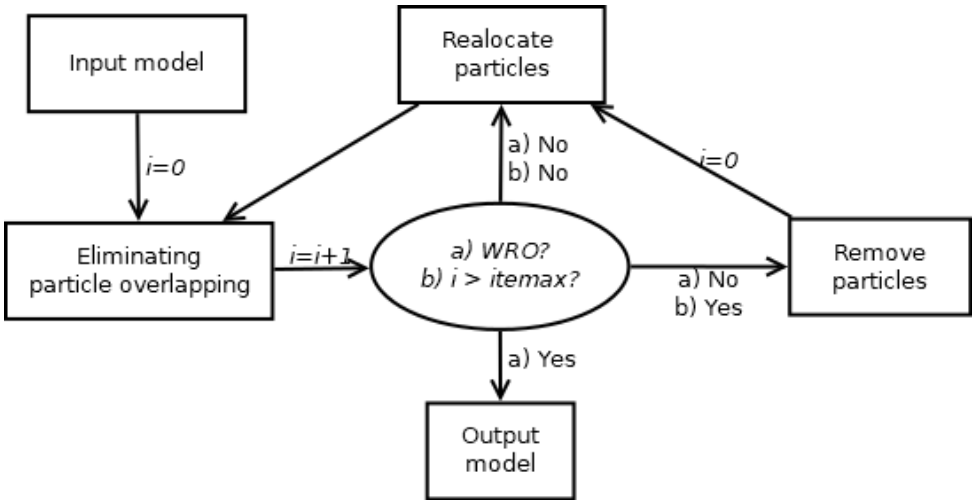
Figure 1 – Macro steps of the work



This work has the objective to develop new distinct particle packing strategies for granular media modeling. The methodologies were studied and worked mainly through these three phases described in the flowchart, each resulting in a different scientific work with its own contributions in the context of particle packing field of study. In the computational development stage, the algorithms responsible for the generation of granular

models are implemented. The particularities in the development, methods used and specific results are presented in subsequent sessions, which deal specifically with each technique. The procedures adopted follow a workflow similar to what is presented in Figure 2.

Figure 2 – General scheme of the proposed methods for modeling particulate media.



The WRO state stands for Well Reduced Overlapping. The methodologies are based on an arrangement of grains in random positions. This initial arrangement is formed from the information about the particle sizes and the index of voids that is intended to be reached in the final model. It is expected the presence of overlapping grains due to the random nature of this initial arrangement. Therefore, the following steps are strategically used so that the geometric overlaps are properly reduced. The strategies presented mainly differ from each other in the phase of overlapping elimination. The technique of distance minimization is used in the Levenberg-Marquardt approach, taking the distances between the faces of the particles as optimization variables. The Geometric Separation method uses an incremental displacement procedure that moves the particles towards the existing overlaps in each step. The alternative GPU version of the Geometric Separation method, despite of mainly following the workflow described in Figure 2, presents very particular methods in order to allow complete parallelization of each step of the methodology. Because this method presented better results in computing time and model size, it is chosen the parallel version of the geometric separation method to be developed and studied in this work. The other procedures, reallocation and removal of particles, are unique and specific for each technique, and the explanations about the particularities of both are described in the specific strategy chapter. After obtaining the computational tools for each proposed strategy, particulate aggregates are modeled from real soil data, demonstrating the applicability and potential of the techniques. The data are extracted from the literature, and the obtained results are compared with the reference. Finally, three scientific papers were produced during the Master's course with the results of each presented strategy described in the following chapters.

1.3 Delimitation

This paper presents particle packing strategies using purely geometric techniques for modeling the granular media. A state of initial stress is not defined in the process of generating the granular model. Depending on the type of application to which the models are intended, it is necessary a further treatment for its use in numerical simulations. Dynamic techniques could be used with the presented strategies to define the initial stress state. These further treatments are not the focus of this work. However, this limitation can be overcome in some applications, such as geomechanical problems related to granular models that the global equilibrium is achieved with the presence of fluids filling the pores between the grains, observed in completely saturated soils or porous rocks. The concepts of method limitations are better described in their respective chapters. The presented strategies focus on two-dimensional approaches, using disk-shaped particles. Considerations of other approaches are also presented in the analyses present in each chapter.

1.4 Document structure

The text structure uses parts of the three scientific articles submitted. Each article is presented in a chapter. Therefore, the document is organized with the following structure:

- The Chapter 2 presents the methodologies and results obtained using the distance minimization particle packing technique. It was submitted to the Engineering with computers Journal;
- The Chapter 3 presents the methodologies and results obtained using the geometric separation methodology. It was submitted to Computational Particle Mechanics Journal;
- The Chapter 4 presents the methodologies and results obtained using the alternative parallel geometric separation with focus in GPU. It was submitted by the Engineering Computations Journal;
- and, finally, the Chapter 5, presents the final considerations and the proposed ways that future works can follow to improve the obtained results presented in this work.

2 A PARTICLE PACKING METHOD FOR NON-UNIFORM SIZES AND PRESCRIBED FILLING RATIO

This chapter presents the development and results of the distance minimization particle packing strategy. The text that follows is based on the scientific paper submitted by Lopes et al. (2019) by the Engineering with Computers Journal. The text corresponds exactly as it is presented in the cited article, with format adaptations to the UFAL dissertation criteria.

2.1 Literature review and initial commentaries

The Discrete Element Method (DEM) (CUNDALL; STRACK, 1979) and other numerical methods have been used as powerful tools to achieve increasingly significant results in the study of engineering problems related to discontinuous, heterogeneous, or granular media. In the recent years, these methods have been applied frequently in fragmentation, fracturing, impact and collision phenomena. A wide use of particle-based methods are directly related to soil modeling in geomechanical problems (CAMPBELL; CLEARY; HOPKINS, 1995; BROWN et al., 2000; ONATE; ROJEK, 2004; CHANG; TABOADA, 2009). Other relevant problems are the fresh concrete compaction and the determination of effective properties for heterogeneous materials (MECHTCHERINE et al., 2014).

Particle-based methods often require the domain discretization on a finite set of particles with well-defined geometric shapes. Usually, circles and spheres are used to represent particles in two- and three-dimensional, respectively. The domain discretization can demand some important requirements: (i) circles or spheres packing with non-uniform sizes; (ii) domain filling with prescribed occupation ratio; (iii) homogeneous distribution of particles; and (iv) global force equilibrium. Despite of the main modeling goals vary according the problem, it is often required the simultaneous meeting of these four characteristics or just some of them combined. In this context, a great number of strategies have been proposed for particle packing, and these techniques can be divided in two distinct groups: the geometric and dynamic algorithms.

Dynamic algorithms use the solution of the motion equation as a mechanism to define the positions of the particles. In solutions by DEM, e.g., the packing algorithms employ similar or identical physical representations to those used in the analysis methodology of the numerical method (LIU; ZHANG; YU, 1999; SIIRIA; YLIRUUSI, 2007). Particulate domains with defined geometric shapes can be generated by dynamic algorithms, using or not non-uniform particle radii. However, a disadvantage of these algorithms is the computational cost demanded. In several cases, the parameters and physical models employed in the particle distribution require the computation of a large number of

iterations. This happens, frequently, due to the stability criteria imposed by the explicit temporal integration algorithms used in the motion equation computation. In some cases, the limiting factor for the packing algorithm efficiency is not just the number of particles involved, but also the number of solving steps demanded by the dynamic solution.

Geometric algorithms are characterized by the use of purely geometric techniques to distribute particles on a given domain. In several cases, this feature significantly reduces the processing time required in the modeling, since it does not need the solution of the motion equations. Cui and O'Sullivan (CUI; O'SULLIVAN, 2003) presented a geometric technique of particle packing based on an initial mesh of triangular or tetrahedral elements. This technique computes the particles positions based on the geometry of the elements and nodes in the mesh. A restricting factor of this technique is the need of an initial mesh of elements and the difficulty on the particle size control.

Another geometrical technique, presented by Frery et al. (FRERY et al., 2011), prescribes the positions of the particles using the theory of point processes aiming to avoid the geometrical overlapping of particles. In this technique, particle sizes are defined initially by the problem characteristics. Next, it tries to put these particles inside the domain until meeting a prescribed filling ratio. In the work, the proposed methodology is applied to soil media, where particle sizes and prescribed filling ratio are represented by grain size distribution curves and porosity, respectively.

Using the concept of advancing front mesh generation, Feng, Han and Owen (2003) present a technique where the positions of the particles inside the domain are calculated based on the previously included particles. The technique is very efficient, but has some limitations in particular cases of three-dimensional models. In these cases the algorithm complexity increases and damages its efficiency. The same authors present a simple geometric technique that is based on compression algorithm (HAN; FENG; OWEN, 2005). In the employed method, particles are distributed at random positions with no overlapping. Next, they are moved properly in a compression direction.

Still in the geometric algorithms context, Oñate and Labra (2009) present a technique of particle packing for dense media. The idea is to change an initial particle arrangement by a process of distances minimization, where the particles are incrementally moved or resized. The procedure uses the Levenberg-Marquardt's minimization strategy (LEVENBERG, 1944; MARQUARDT, 1963), which is also employed in this work. However, the work proposed by Labra and Oñate does not ensures both prescribed particle sizes and domain filling ratio.

In summary, dynamic algorithms usually allows to generate models with prescribed particle sizes and global force equilibrium. However, the computational cost of these algorithms can make prohibitive their use, especially on large-scale models. Possible solutions to improve the computing time is presented by Baugh and Konduri, and Cintra

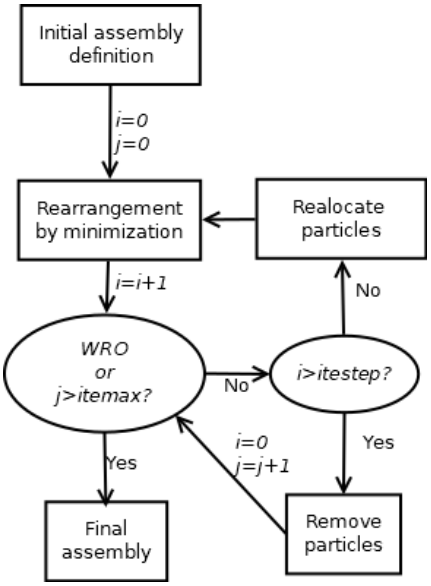
et al. (BAUGH; KONDURI, 2001; CINTRA et al., 2016a). In their work, they show that it is possible to reduce the convergence time of large DEM simulations using High-Performance Computing (HPC). On the other hand, traditional geometric algorithms have difficulty to achieve responses that meet the specified particle sizes and the dynamic equilibrium. However, geometric techniques can be used in certain types of problems, such as fully saturated soils, as described by Frery et al. (FRERY et al., 2011). Frery proposes searches to meet both filling ratio and size distribution of real granular models with spatial homogeneity, achieving close results to experimental data of fully saturated soils.

Considering the geometric algorithms background, this paper presents an alternative procedure to distribute circles with non-uniform sizes and a prescribed filling ratio inside an arbitrary domain. The proposed method searches to reproduce characteristics of real granular models, such as soils. The adopted methodology also ensures homogeneity for the spatial distribution of the granular set. More specifically, this paper proposes a variant for the strategy presented by Labra and Oñate (LABRA; ONATE, 2009). The developed changes in this work are motivated because the method presented by Oñate and Labra are applied to dense granular media, and it does not search to meet prescribed characteristics of granular models. However, due to the set of restrictions and changes to meet the characteristics of the granular models, additional treatments are conduced to reduce the computing time and approx the final models to real particulate aggregates. The examples presented by Frery et al. (FRERY et al.) are reproduced with the proposed method, and results are compared with each other and the real soil data in order to validate the strategy.

2.2 PROPOSED STRATEGY

In order to achieve the proposed objectives, the adopted strategy is based on three macro steps, as can be seen in Fig. 3. The first step of the procedure is to obtain an initial set of particles, and the particle size and filling rate are the input data required to generate this initial configuration. Particles are then inserted into random positions of the domain from this input data, respecting the particle size distribution and doamin filling ratio. Due to the random nature of this initial assembly, it is expected overlapping between each other particle and between particles and the borders of the domain boundaries. The next steps are strategically used and repeated to reduce these overlaps: i) rearrangement by minimization; ii) reallocation of particles; and iii) removal of particles. The reallocation step changes the position of the particles aiming to minimize the overall overlaps. If the overlaps are not sufficiently reduced after several attempts, some particles are removed from domain. However, this particle removal is not desired because it affects the output configuration of both filling ratio and particle size proportion.These steps described in Fig. 3 are presented with details in the following subsections.

Figure 3 – Adopted strategy

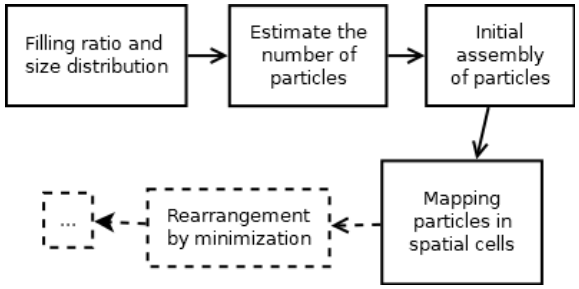


This strategy aims to reach a state of Well Reduced Overlapping (WRO). In this state, both particle-particle and particle-boundary overlapping are satisfactory reduced. The index i indicates a counter for the number of iteractions in which the particle reallocation fails. If the counter i exceeds $itestep$ (the maximum number of iteractions per minimization step), it is necessary to remove particles. The index j is a counter for the number of iteractive steps of distance minimization. If the counter j exceeds $itemax$, the method achieve the maximum number of iteractions and does not meet the target parameters for the final model.

2.2.1 Initial assembly definition

The input data for the initial assembly definition are the filling ratio of solids and grain size distribution. The flow of this step is shown in Fig. 4. The initial number of particles is estimated from this information, as described in Section 2.2.1.2, and those particles are inserted one by one into the domain at random positions.

Figure 4 – Initial assembly definition



All generated particles are mapped in spatial cells using the Non-binary Searching (NBS) algorithm (MUNJIZA; ANDREWS, 1998). This subdivision of domain by cells

helps to search for overlaps using the particle neighborhood instead of the entire domain.

2.2.1.1 Filling rate and size distribution

Filling ratio and size distribution are important data related to granular media. A numeric value can be described as shown in the Eq. (2.1) , where S_R represents the filling ratio, also known as solid ratio.

$$S_R = \frac{V_s}{V}, \tag{2.1}$$

where V is the volume of the domain, and V_s is the volume filled by solids. In soils, this filling rate generally have an explicit relationship with porosity. It can be computed as the ratio between the empty volume (not filled by grains) and the total volume, as described:

$$\eta = \frac{V_v}{V}, \tag{2.2}$$

where V_v is the volume filled by voids, and η represents the porosity. The focus of this work is to reproduce bi-dimensional problems. Therefore, the solid and empty volumes must be represented by the equivalent areas. This conversion is made, for the case of circular particles, considering each circle as a cylinder of unitary height. The equivalence between volumes and areas follows according to the expression:

$$V^* = A^* h, \tag{2.3}$$

where h is the unitary height, and A^* is the equivalent area related to the volume V^*. The filling ratio is important to define the porosity η in a domain, as described in Eq. (2.4).

$$S_R = 1 - \eta. \tag{2.4}$$

A size distribution for a collection of grains could be defined by discrete ranges of particle sizes, or grain size fractions. It is possible to estimate the total number of particles of a model combining both size distribution and filling ratio. This must attend the input data, the following approximation needs to be respected:

$$(1-\eta)V \approx \sum_{i=1}^{g} w_i (1-\eta)V, \tag{2.5}$$

where w_i represents the occupation fraction for the i-th grain size fraction, V is the representative solid volume, g indicates the number of fractions and $(1-\eta)$ is the model's filling ratio. w_i is the area ratio of each fraction, its value must be positive and need to meet:

$$\sum_{i=1}^{g} w_i = 1. \tag{2.6}$$

The size distribution for a general granular model is defined by the set of diameters d_i, and its respective fraction in the total of particles.

2.2.1.2 Estimating the number of particles

The total number of particles can be estimated using the input data and the Eq. (2.5). Despite the numerical value of the filling ratio being in a continuous range, the total number of particles is an integer number. Therefore, the value of n calculated in the Eq. (2.5) must be truncated. An inherited numerical error is a natural consequence of the discontinuous nature of the granular model and the truncated value. This error is minimized when increasing the number of particles. The general equation shown in Eq. (2.5) can be modified to be applied in different cases of interest. For instance, the Eq. (2.7) represents the same relation applied to a model that uses cylinders with diameter d_i and hypothetical unitary height:

$$n = \sum_{i=1}^{g} \frac{4(1-\eta)Vw_i}{\pi {d_i}^2}, \tag{2.7}$$

or, in the case of a single grain size fraction:

$$n_i = \frac{4(1-\eta)Vw_i}{\pi d^2}. \tag{2.8}$$

For the general case of spherical particles, the number of particles can be estimated in a similar way:

$$n = \sum_{i=1}^{g} \frac{8w_i(1-\eta)V}{\pi {d_i}^3}. \tag{2.9}$$

The inherited error inserted n the calculus of the total number of particles is also inherited in the calculus of the number of particles in each grain size fraction. This error could be expressed according to:

$$n_i = floor(\frac{(1-\eta)Vw_i}{\pi v_i}), \tag{2.10}$$

where (for circles case),

$$v_i = floor(\frac{\pi {d_i}^2}{4}), \tag{2.11}$$

or (for spheres case),

$$v_i = floor(\frac{\pi {d_i}^3}{8}), \tag{2.12}$$

where $floor$ is a mathematical operator that returns the integer part of a real number. It is possible to measure the error of the entire set in terms of the filling ratio residue. This value can be calculated relatively to the expected filling ratio S_e:

$$\epsilon = 1 - \left(\frac{V - \sum_{i=1}^{g} n_i v_i}{(1-S_e)V} \right). \tag{2.13}$$

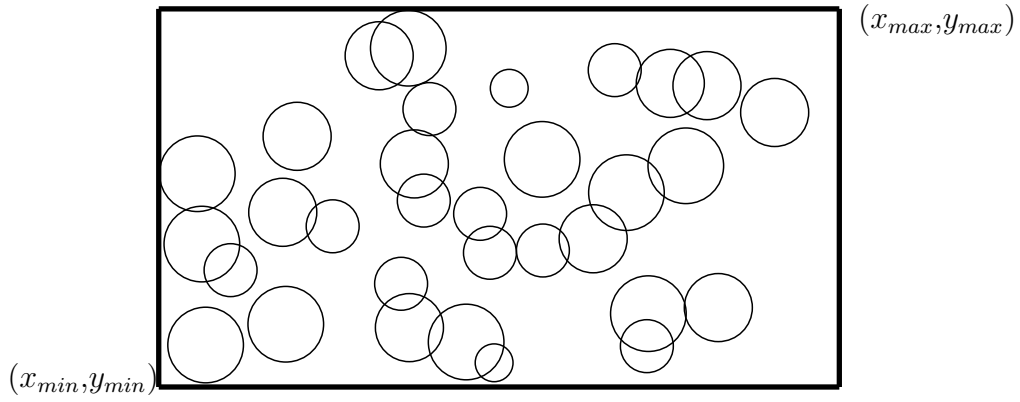
Previous analyses had shown that models composed with particles of great size also have higher filling ratio residues, while models with smaller particles presents less error. These are characteristics of discrete models, in which increasing the discretization means

to decrease the error. Despite the description of the method so far have represented mathematically the maximum quantities of particles to be inserted, nothing still was considered about the geometric shape or the spatial arrangement (positions). As the initial assembly is obtained by inserting the particles in a random position, and it does not consider the overlaps, the geometric shape and positions is only considered in the following steps. The random generation with no overlapping restriction is shown in Fig. 5.

2.2.1.3 Initial particle distribution in the domain

This step consists in a random distribution of particles in the domain from the quantities estimated in the subsection 2.2.1.2. Fig. 5 illustrates an initial particle assembly.

Figure 5 – Initial particle assembly in a bi-dimensional example, showing overlapping between particles.

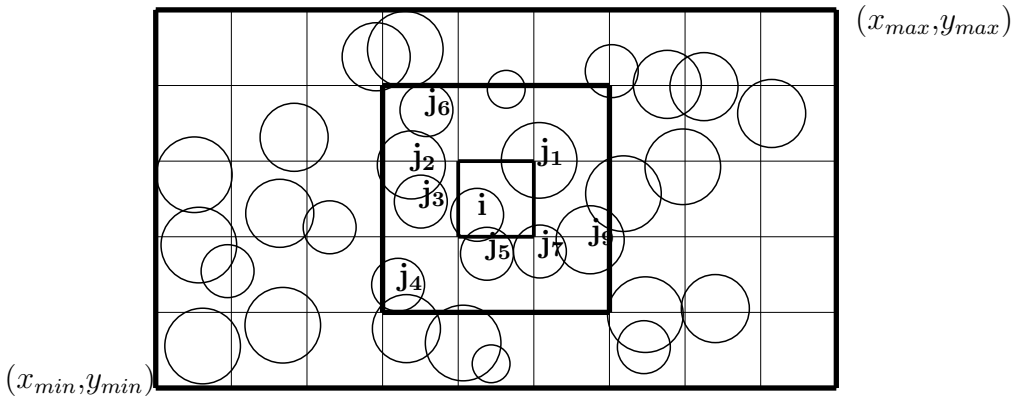


The particle coordinates are randomly set in the domain one by one, respecting the radii proportion and quantities. There are no constraints for the particle coordinates, therefore some of them overlap each other. The strategy aims to mitigate these overlapping along the iteractions.

2.2.1.4 Mapping the particles in spatial cells

All particles are mapped into spatial cells that represent subdivisions of the domain. This mapping is important for the efficiency of the algorithm and it is a prerequisite for the rearrangement procedure by minimizing the distance of particles, as described in the section 2.3. The main goal is to reduce the searching for contacts or incidences to a near neighborhood, as described in section 2.2. The strategy consists in creating a cell structure that contains the data of the particles that are inside the respective spatial cells. A mapping algorithm is executed in order to create the initial cell structure, setting which particles are inside of each cell. This algorithm is used many times during the generation, due the positions update. Fig. 6 illustrates the cell division. The bold smaller square is the cell that contains the particle "i", and the larger bold square represents the immediate neighborhood that contains the particles "j_k". Searching for particles and incidences in

Figure 6 – Domain divided by cells. The external bold square contains the j particles in the immediate neighborhood for the particle i, and the internal bold square is the cell that contains the particle i.



near neighborhoods is computationally more efficient than searching through the entire domain.

The literature presents techniques of cell mapping often adopted in Discrete Element Method simulations (MUNJIZA; ANDREWS, 1998; WILLIAMS; PERKINS; COOK, 2004). This work uses the classic algorithm NBS used by Munjiza and Andrews (MUNJIZA; ANDREWS, 1998), in which each cell is a square space that the edges measure two times the larger radii size of all particles. In general, searching algorithms are conceived and studied to this default cell size. Cells with other sizes may not work properly, presenting nonconformity with the standard setting.

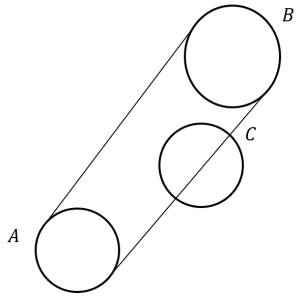
2.2.2 Definition of incidences

An important step for the distance minimization algorithm is the definition of incidences. The mapping of these incidences is executed inside each cell, and they can be divided in two types: contact and visibility. A contact incidence occurs when two particles overlap each other. This happens when the distance between the particle centers is inferior to the sum of their radii. The visibility incidence happens when there is nothing, particles or obstacles, on the area delimited by the outer tangent lines between two particles. In the Fig. 7, the particles A and B are not visible to each other because the presence of the particle C. Minimizing the distance between visible particles tends to reduce the amount of empty gaps in models, thus helping to maintain homogeneity. Only the distance between two incident particles will be considered in the distance minimization step. The contact incidences are also used when removing and reallocating particles.

2.2.3 Rearrangement by distance minimization

Rearranging particles by distance minimization is the main method used to change their positions. This process aims to eliminate the overlapping between each other particle

Figure 7 – Visibilty test.



and the particles and the domain boundary. The strategy used is an adaptation of the method proposed by Labra and Oñate (LABRA; ONATE, 2009). The original idea of this authors considers both the positions and diameters of the particles as decision variables in the distance minimization process. The present work proposes to consider only the particle positions as decision variables of the minimization.The diameters are kept unchanged relative to the initial assembly in order to achieve targets grain size distribution and filling ratio. The variables involved into the minimization are the n positions of the particle centers, which n is the total number of particles. So, the global minimization will return as result the following set:

$$\boldsymbol{x} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_i, ..., \boldsymbol{x}_n\}. \tag{2.14}$$

In this set, x_i represents the spatial coordinates of the particle i, and could be express as:

$$\boldsymbol{x}_i = \{x_i, y_i\} \quad \text{(bi-dimentional particles)}, \tag{2.15}$$

or,

$$\boldsymbol{x}_i = \{x_i, y_i, z_i\} \quad \text{(tri-dimentional particles)}. \tag{2.16}$$

Labra and Oñate proposed a strategy that creates dense models with high values of filling ratio. There is no need to prescribe an input ratio or grain size distribution, because these values are not fixed and can be changed to achieve the densest particle package possible. The strategy proposed in the present work takes an alternative approach, obtaining models with specified filling ratio and grain size distribution. The computing time is compromised due this change, because more iteractions of the minimization algorithm will be needed to achieve the final model. Section 3 presents a set of methods that are used to reduce the processing time. The approach to get the final arrangement of the granular model involves obtaining configurations of x that minimize the sum of square distances between particles into a spatial neighborhood (incidences).The minimization problem have following objective function::

$$\min f(\boldsymbol{x}) = \sum_{i=1}^{n} p_i, \tag{2.17}$$

for

$$p_i = \sum_{j=1}^{n} \Phi_{ij} E_{ij}, \tag{2.18}$$

and,

$$E_{ij} = \left[||\boldsymbol{x}_i - \boldsymbol{x}_j|| - (r_i + r_j) \right]^2. \tag{2.19}$$

Where Φ_{ij} is a matrix which the elements have values equal to one for each incident pairs $[i, j]$ and null for the rest. The radius of the particles are constant in the minimization problem and are represented as $[r_i, r_j]$. The adopted solution for Eq. (2.17) is the Levemberg - Marquardt scheme (LEVENBERG, 1944; MARQUARDT, 1963). The objective variables inside de x set are increased iteractively as such:

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \boldsymbol{h}^k, \tag{2.20}$$

in which the increment $\boldsymbol{h}^k$ is the linear system solution:

$$\left[\boldsymbol{J}^T \boldsymbol{J} - \mu \boldsymbol{I} \right] \boldsymbol{h}^k = -\boldsymbol{J}^T \boldsymbol{p}. \tag{2.21}$$

$\boldsymbol{I}$ identity matrix and $\boldsymbol{J}$ is the Jacobian matrix, given by:

$$J_{ij} = \frac{\partial p_i}{\partial \boldsymbol{x}_j}. \tag{2.22}$$

The scalar value μ is a smoothness parameter of Levemberg - Marquardt. This Parameter must be positive to assure that the increments obtained are in a descendent direction. The iteractive process described in Eq. (2.20) must be repeated until the increments $\boldsymbol{h}^k$ assume values lower than a tolerance or when the maximum number of iteractions are exceeded. It is advised the storage in spatial data structures, especially in large models, of the both matrices J_{ij} and the coefficients of the linear equation system shown in Eq. (2.21).

2.2.4 Reallocating particles

An important step to reduce the computing time demanded to eliminate the overlaps is the particle reallocation. The particles presenting most overlapping will be reallocated to regions with low values of filling ratio. This can be achieved using the cell structure proposed in Section 2.2.1.4. Cells which there is no particle centers inside are called empty cells. These empty cells are used to identify empty regions. Particles presenting most overlapping will be moved to fill the porous region and rising the homogeneity of the model. The rearrangement by minimization is repeated until all the empty cells are filled. This step will not be executed if there are no empty cells into the domain. The searching for empty cells is influenced by the cell sizes. As mentioned in Section 2.2.1.4, where each cell is a square space that the edges measure two times the maximum radii size of all particles. This guarantee a better convergence in the minimization algorithm, because the

larger particle still fits totally in each cell. However, even if just one particle much smaller than cell size is inside, this cell is not considered empty and presents. A case of study using half of the cell size reduces significantly the filling ratio error, but the computing time demanded increase. For instance, the use the half of default cell size results in up to 6 % error reduction for the tested examples, and also making the generation up to five times slower.

2.2.5 Removing particles

A particle removal is realized for particles with many overlapping, similar to the particle reallocation procedure. The number of particles to be removed must respect a value called *porosity increment* (Δ_η). This value must be small (between 0.01% and 1%) to guarantee that the output filling ratio value is really close to that presented in the input model. Therefore, a certain area of particles must be removed respecting this increment of filling ratio:

$$A_R = A\Delta_\eta. \tag{2.23}$$

Particles removed will respect an area that does not surpass A_R. The value of the final filling ratio must be also updated according to the number of removed particles. This process will be repeated, along with the reallocation, until the reduction of the overlaps achieves the desired tolerance. Removing a particle harms the achievement of the target parameters defined by the initial model, because the total number of particles to get that goal is firstly generated in the initial assembly. Even respecting the filling ratio increment, the process of removing particles may be repeated several times. That only could increase the porosity of the model to values far beyond that increment. Due this respect, some techniques to reduce the error are discussed in the next subsection.

2.3 Numerical studies

This section presents the numerical results and contributions achieved in this work. Several examples are used to calibrate the strategy, and solutions for the limitations are discussed.

2.3.1 Filling ratio error and response surface

The proposed strategy presents a filling ratio error when comparing the reached ratio value and that used in the initial assembly. This occurs, mainly, due to particle removing from domain. The value obtained is usually smaller than expected. Some measures are applied to overcome this matter. One of the approaches is to make the input value of filling ratio larger than the targeted one. This creates an initial assembly with more particles than the targeted model would need, and it allows strongger removing of overlapping particles.

Examples provided by Frery et al. (FRERY et al., 2011), are used to generate a response surface in order to know the approximated value of filling ratio that mitigate the error in the final model. Frery et al. presented seven examples of granular models generated from different soils. Because the samples are related to soils, the value of porosity is used to characterize the models instead of the filling ratio, as described in Section 2.2.1.1. The data provided by a sieve analysis is represented by probabilistic log-normal distributions, as shown in Table 1.

Table 1 – Probabilistic normal distribution of soil samples (FRERY et al., 2011)

Sample	μ	σ
1	2.338	0.159
2	2.366	0.054
3	2.393	0.113
4	2.354	0.051
5	2.274	0.095
6	2.289	0.106
7	2.403	0.055

These probabilistic distributions are used to generate a set of particles with diameters in the Phi scale of Krumbein (KRUMBEIN, 1937). The diameters of the grains are measured in this scale and expressed by the equation:

$$\Phi = -\log_2(\frac{D}{D_o}), \tag{2.24}$$

where D represents the grain diameter, and D_o represents a reference diameter, usually equals to 1 mm. Several numerical models are generated based in the grain size distributions of the samples, varying porosity and domain size. A response surface is generated to mitigate the porosity error and it is represented by a polynomial mathematical function. This surface is obtained from several tests and error results. The multivariables of this polynom are input porosity, a parameter named size factor, and the inclusive graphic standard deviation (UDDEN, 1914). The size factor can be calculated as described in Eq. (2.25) :

$$sf = \frac{\sqrt{Area_{domain}}}{R_{mean}} \tag{2.25}$$

Where R_{mean} is the mean radius between the grains.

$$R_{mean} = \sum_{i=0}^{g} w_i R_i \tag{2.26}$$

This measure of size factor is defined for square and rectangular domains. However, examples are used to evaluate others convex domains, showing close results in both cases.

The inclusive graphic standard deviation measures the homogeneity of the grain size distribution of a soil. It is computed by:

$$\sigma_1 = 0.25(\Phi 84 - \Phi 16) + 0.15(\Phi 95 - \Phi 5). \tag{2.27}$$

A response surface that fits the input data and error values of the generated granular models is described in Eq. (2.28) and shown in Fig. 8. Fig. 8 shows three different configurations of the surface varying the graphic standard deviation. The error results for this surface are shown in Fig. 9. The values of the coefficients A_i, B_i, C_i, D_i, E_i and F_i are discriminated in Table 2 .

$$\begin{aligned}\eta_f = \sum_{a=0}^{3} A_a \eta^a + \sum_{b=0}^{3} B_b sf^b + \sum_{c=0}^{2} E_c 2^{\sigma_1^c} + \\ \sum_{i=0}^{3} C_i \eta^i . \sum_{j=0}^{3} D_j sf^j . \sum_{k=0}^{1} F_k 2^{\sigma_1^k}\end{aligned} \tag{2.28}$$

η_f represents the final porosity obtained by the response surface.

Table 2 – Response surface coefficient values.

index	0	1	2	3
A	$-1.38E6$	0.53	0.61	-0.28
B	$-1.62E7$	$-4.32E$-3	$7.16E$-5	$-2.55E$-7
C	0.68	0.18	-0.85	0.61
D	0.67	0.7	0.28	$2.96E$-3
E	$1.76E7$	-0.6	-0.58	
F	$-1.63E$-5	$-3.97E$-5		

Table 3 – Grain size distribution from sample 2

Φ	**Proportion**
0	0,01%
1	1,38%
2	28.66%
3	20,90%
4	38,42%
5	10,59%
6	0,04%

Samples presented by Frery are composed by silt, clay and sand, that comprehend mostly values of Φ from zero to eight. It is achieved a grain size distribution through the

Figure 8 – Response surface for different values of graphic standard deviation.

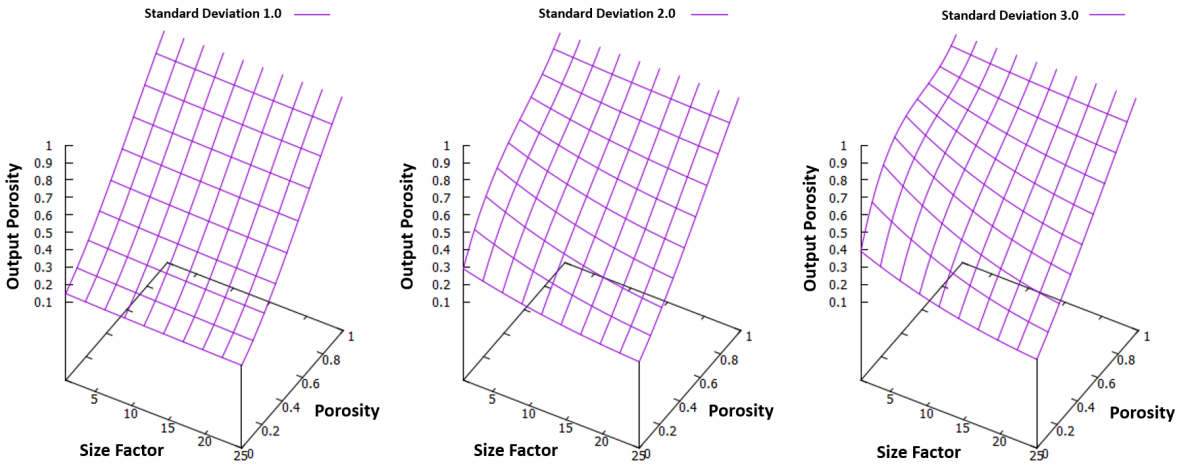


Table 4 – Error by grain size fraction varying porosity values

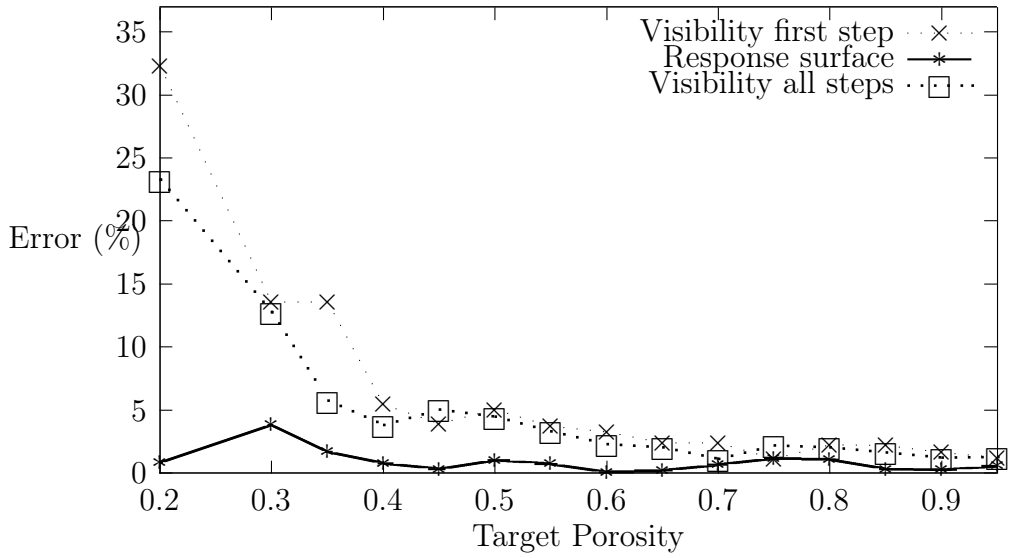
Porosity	0ϕ	1ϕ	2ϕ	3ϕ	4ϕ	5ϕ	6ϕ
0.2	0.0041%	0.294%	1.288%	0.916%	1.104%	1.38%	0.01%
0.25	0.0012%	0.494%	1.565%	1.475%	2.142%	0.00138%	0.01%
0.3	0.006%	0.179%	1.175%	0.855%	0.813%	0.00138%	0.01%
0.4	0.003%	0.22%	0.877%	0.466%	0.173%	0.00138%	0.01%
0.45	0.005%	0.054%	0.815%	0.248%	0.279%	0.00138%	0.01%
0.5	0.002%	0.078%	0.456%	0.36%	0.497%	1.38%	0.01%
0.55	0.008%	0.288%	1.825%	0.426%	1.142%	1.38%	0.01%
0.6	0.004%	0.112%	1.7%	0.6%	1.018%	1.38%	0.01%
0.65	0.001%	0.274%	1.76%	0.064%	0.709%	1.38%	0.01%
0.7	0.009%	0.294%	1.341%	0.239%	0.475%	1.38%	0.01%
0.75	0.003%	0.221%	0.973%	0.425%	0.226%	1.38%	0.01%
0.8	0.017%	0.013%	0.945%	0.298%	0.772%	1.38%	0.01%
0.85	0.008%	0.98%	2.139%	1.979%	3.701%	1.38%	0.01%
0.9	0.007%	0.849%	2.086%	3.104%	4.656%	1.38%	0.01%
0.95	0.04%	0.521%	4.024%	1.322%	4.438%	1.38%	0.01%

parameters of the second sample. This distribution is presented in Table 3 in the Phi scale of Krumbein.

Following tests are the porosity error for models generated using this size distribution and varying values of porosity in the initial model. Fig. 9 shows the relationship between the target porosity and the error (%) for three different approaches.

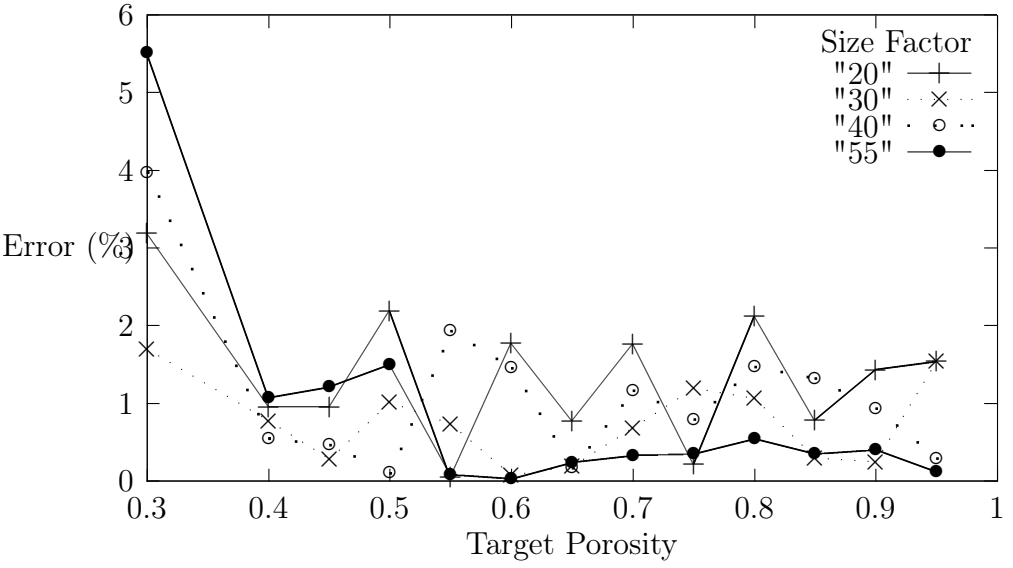
The response surface method presents the better error results when comparing to the method without its uset. Visibility incidences are only considered in the first iteraction of this method in order to reduce the void gaps and only contact incidences are considered

Figure 9 – Porosity error using the size distribution of the sample 2.



afterwards. Studies realized show that is only possible assure, with an error margin lower to 7%, models with porosity upper to 18%. The use of visibility incidences in all interactions could reduce the error, but the speed of the model generation is compromised. Model generations with the visibility algorithm used beyond the first distance minimization could be slower due to the additional operations needed. The response surface is designed to an interval of box sizes varying from 10 to 100 times the largest particle radius. Regarding to non-square boxes, the size factor is used a normalized value that considers both the size of the domain and particles. Fig. 10 shows that the size factor have few influence in the behavior of the target porosity error.

Figure 10 – Porosity error for different size factor values.



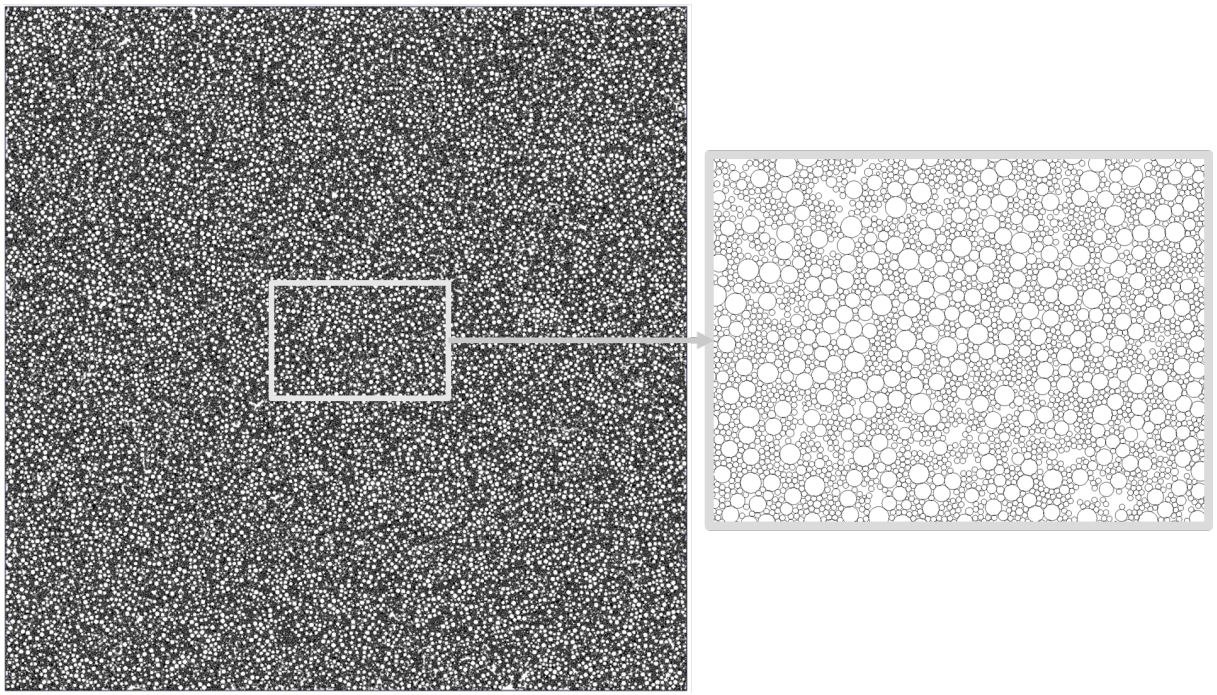
2.3.2 Grain size distribution error

One of the main goals of the proposed method is reaching a target grain size distribution. This means that the proportion between the radii in the final assembly must be equal or at least close to the input size distribution. An investigation of this aspect is done by evaluating of the number of particles, in each grain size fraction, of the input model and of the final model. Table 4 shows the results of this grain size fraction error for the Fery etl al. Sample 2.

The maximum error value shown in table is lower than 5%, and the mean error value is close to 1 %. These results show that the input size distribution is respected closely in the final model.

2.3.3 Examples

Figure 11 – Model with 113,154 particles generated.



Granular models are used in order to illustrate the applicability of the strategy. The first example is a box that represents a soil sample. Its dimensions are 900x 900, and the grain size fractions respects a log-normal distribution used to generate the response surface and porosity of 20%. The initial assembly is obtained with the estimated number of particles described in Section 2.2.1.2 and the response surface, presenting a total of the model has 115,679 particles. The final model presents 113,154 particles, due the particle removal, as shown in Fig. 11, and its porosity value achieved in the final model is 20.545%. This result shows that, despite the number of deleted particles, the model obtained by the proposed method presents a close porosity value in comparison to the target one.

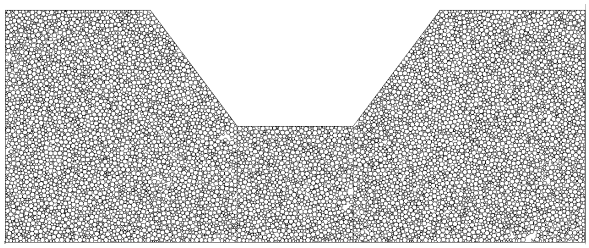
Table 5 – Data profile of the example 1.

Radius	**target**	**Output**
3.60	1.45 %	1.445 %
3.00	4.79 %	4.792 %
2.6	1.86 %	1.867 %
2.20	1.63 %	1.633 %
1.80	2.43 %	2.447 %
1.40	1.94 %	1.941 %
1.00	83.11 %	83.034 %
0.60	1.00%	1.007 %
0.30	1.75%	1.750%
Porosity	**20%**	**20.545%**
Particle Number	**115332**	**113296**

The Table 5 shows the data profile for the generated example, in which both the size distribution and target porosity are achieved according the expected tolerance.

The methodology is not applicable to generations on non-convex domains directly without further treatments for overlapping that occurs in concave corners of the polygons. However, the merge of particle sets generated in convex domains, can allow to fill several arbitrary non-regular polygon. Fig. (12) shows the particle generation on a non-convex domain composed by two regular pentagons and a square. It is important to emphasize that additional treatments on particles near the polygonal interfaces may be necessary to ensure the required degree of continuity in some problems. The target porosity is the same used in the last example, as well as grain size fractions. However, the grain size distribution is uniform. Table 6 presents the profile of the final assemble, and it shows good correspondence with the target values.

Figure 12 – Particle packing inside a non-convex domain.



The seven soil sample distributions presented in the Frery's work (FRERY et al., 2011) are reproduced as granular models for validation of the present strategy. The results obtained, as well as the grain size distribution profiles achieved with the proposed strategy are presented in Tables 7 and 8. In table 26, η represents the porosity value of the real particulate aggregates, η_{Frery} corresponds to the results presented in Frery's work, and n' represents the results obtained from proposed strategy. The values are close to the real results and those obtained by Frery. Results presented in table 7 show that samples 1, 3

Table 6 – Data profile for the example 2.

Radius	**Target**	**Output**
0.50	20 %	19.74 %
0.40	20 %	20.00 %
0.30	20 %	20.13 %
0.20	20 %	20.01 %
0.10	20 %	22.12 %
Porosity	**20%**	**20.35%**
Particle Number	**11715**	**11518**

Table 7 – Error by grain size fraction for the samples presented by Frery

Sample	0ϕ	1ϕ	2ϕ	3ϕ	4ϕ	5ϕ	6ϕ	7ϕ	8ϕ
1	0.119%	1.147%	3.433%	5.853%	4.064%	1.801%	1.443%	9.070%	7.790%
2	0.000%	0.000%	0.003%	0.289%	0.640%	0.343%	0.121%	1.380%	0.010%
3	0.000%	0.085%	1.072%	5.495%	6.487%	3.253%	4.938%	10.480%	6.850%
4	0.000%	0.000%	0.001%	0.185%	0.446%	0.129%	0.269%	0.480%	0.010%
5	0.065%	0.114%	0.130%	0.092%	0.024%	0.151%	0.116%	0.073%	0.154%
6	0.002%	0.144%	1.060%	2.673%	1.617%	0.520%	2.527%	3.030%	0.460%
7	0.000%	0.000%	0.000%	0.210%	2.308%	2.069%	2.613%	7.010%	0.190%

and 7 presented an error value greater than 5 % for some of the grain size fractions. These three samples have the highest standard graphic deviation values (σ_1), characterizing soils with a less homogeneous particle size distribution, or poorly graded. Therefore, the porosity of all the generated granular models show close values to the laboratory samples.

Figure 13 – Time demanded to achieve a target porosity with different values of size factor.

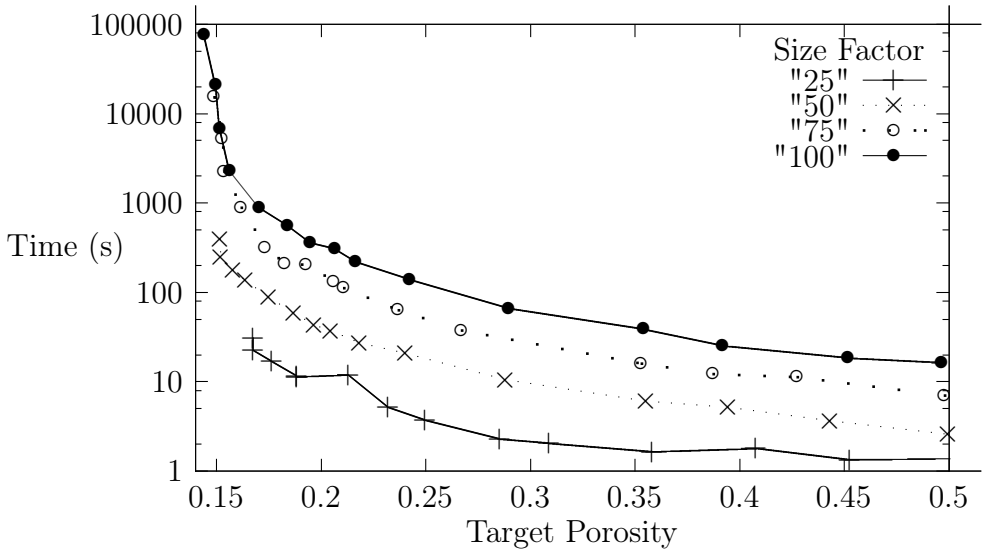


Table 8 – Results for porosity values of soil samples and numerical models.

Sample	η	η_{Frery}	η'
1	0.35	0.345	0.346
2	0.430	0.421	0.427
3	0.420	0.418	0.422
4	0.390	0.390	0.381
5	0.400	0.399	0.392
6	0.330	0.328	0.340
7	0.440	0.430	0.448

2.4 Computational performance

Preliminary studies of the strategy show that some tasks take more time to be finished than others. A critical aspect of the method is the matrix operations demanded by the distance minimization algorithm and the definition of the incidences. In this context, a maximum number of minimization operations are set in each iteraction, and the visibility algorithm to find incidences is only used in the first step. Using the visibility algorithm only in the first step makes model generation up to five times faster. An important goal of the presented strategy is trying to reduce the computing time took for the particle generation process. This time is related to both the number of particles and the target porosity. The Fig 13 shows the relationship between the number of particles and the computing time that the particle generation take in a model with size factor equal to 100 (the upper size factor limit assured for the response surface). All time measurement simulations were performed on a computer containing a four-core Intel Core i5 processor at a clock speed of 3.2 GHz, and the amount of available ram memory is 16 GB. The figure shows that the computing time is directly proportional to both the size factor and target porosity. The generation time ranges between 100 and 1000 seconds on models with target porosity above the 18%, which is the interval of the examples generated in this work. However, the results show that it is necessary over than 20 hours to generate a model with 14.41% of output porosity. Despite being a plausible time span when speaking of DEM simulations, it is above the time expected to generate granular models.

2.5 Commentaries about the technique

This work presented a particle packing algorithm with non-uniform sizes and prescribed filling ratio. The final particle assembly is obtained by minimizing the distances of the particle centers that are initially generated in random positions. A response surface is also presented in order to minimize the error associated to the target filling ratio and that obtained of final assembly. This enables better results without compromising compute time. A natural evolution of the algorithm involves a three-dimensional version of the

strategy. Initial studies have shown good correspondence between the 3D strategy and that presented in the present work. However, studies are still necessary to achieve a better understanding in the correlation of the parameters used, e.g. the size factor and the filling ratio. The computing time and memory demanded for the 3D algorithm is also far greater than the two-dimensional problem. The present strategy achieves the main goal of generating of particle arrangements with prescribed filling ratio and grain size distribution. Studies in the computational cost of the generated samples show that parallel strategies could be used to improve the the convergence time (BAUGH; KONDURI, 2001; CINTRA et al., 2016a).

3 A NON-UNIFORM CIRCLE PACKING METHOD USING GEOMETRIC SEPARATION

This chapter presents the development, results and performance tests obtained through the geometric separation particle packing methodology. The text that follows is based on the scientific paper submitted by Lopes, Cintra and Lira (2019a) by the Computational Particle Mechanics Journal. Some aspects of the methodology are similar to that presented in Chapter 2, especially the generation of the input model, and are again explained here. Other new concepts are introduced and several examples, technique validation and comparison of results with the method of the previous chapter and with the literature are shown. The text that follows is in the form of a scientific paper already submitted, containing the necessary changes to meet the scope of the dissertation format.

3.1 Literature review and initial commentaries

Considering their large applicability, particle based numerical methods have often been applied to solve engineering problems that involve discrete media, such as fragmentation, fracturing, impact and collision phenomena. A wide use of particle-based methods is also directly related to soil modeling in these geomechanical problems (CAMPBELL; CLEARY; HOPKINS, 1995; BROWN et al., 2000; ONATE; ROJEK, 2004; CHANG; TABOADA, 2009; CLAUSEN et al., 2019) and other relevant problems, such as the compaction of fresh concrete and the determination of effective properties for heterogeneous materials (MECHTCHERINE et al., 2014; CAMPOS et al., 2019).

An important task in DEM simulations is the generation of an initial particle assembly that approximately represents a real granular model. A discrete media model can often demand to attend some important requirements: (i) particles (disks or spheres) packing with non-uniform sizes; (ii) domain filling with prescribed occupation ratio; (iii) homogeneous distribution of particles; and (iv) global force equilibrium. The literature presents a variety of methods for particle packing, mainly for circular and spherical particles in the bi- and tri-dimensional models, respectively. Although the characteristics of the output model may vary according to the problem, it is often necessary to simultaneously meet these four characteristics or some of them combined. In this context, several approaches have been proposed to obtain an initial packaging of particles that meet the desired requirements. This work focuses on a category of methods known as geometric packing algorithms. These geometric algorithms define the initial positions of particles based on geometric procedures, using the characteristics of the particles and of the domain. The movement of particles during DEM simulations obeys physical laws and equations of motion. It is possible to build the initial set of particles using the same formulations and procedures used by the

DEM simulations. The algorithms that use this formulation are called dynamic algorithms. In several cases, the use of geometric methods significantly reduces the processing time required in modeling, since it does not need the solution of the equations of motion.

Cui and O'Sullivan (CUI; O'SULLIVAN, 2003) presented a geometric particle packing technique based on an initial mesh of triangular or tetrahedral elements. This technique calculates the position of particles based on the geometry of the mesh elements and nodes. A limitation of this technique is the usage of an initial element mesh and the difficulty in controlling the particle sizes.

Feng et al. (FENG; HAN; OWEN, 2003) present a technique based on advancing front, where the positions of the particles on the domain are calculated based on the previously included particles. The technique is very useful, but has some limitations on three-dimensional domains, impairing its efficiency. The same authors present another geometric strategy based on the compression algorithm (HAN; FENG; OWEN, 2005). In this method, the particles are distributed in random positions with no overlapping and they are properly moved in a compression direction.

Frery et al. (FRERY et al., 2011) proposes a method that prescribes the positions of particles by the theory of point processes. The technique avoids geometric overlapping of particles, and particle sizes are initially defined by the characteristics of the problem, such as the grain size distribution and domain filling ratio. It then tries to place these particles on the domain until a prescribed filling rate is achieved. In the examples presented by Frery, the methodology is applied to problems involving the soil, where the particle sizes and the prescribed filling ratio are represented by grain size distribution and porosity, respectively.

Oñate and Labra (LABRA; ONATE, 2009) propose a particle packing method to model dense media using a minimization strategy. The idea is to change an initial arrangement of particles using a distance minimization process, where particles are incrementally moved or resized. The procedure uses the Levenberg-Marquardt's minimization strategy (LEVENBERG, 1944; MARQUARDT, 1963). However, this work does not focus in attending prescribed particle sizes and domain filling ratio.

Lozano et al. (LOZANO et al., 2016) proposes an efficient algorithm that is capable of producing packs with millions of spheres following a statistical sphere size distribution inside complex arbitrary domains. This work uses a seed boundary advance method and a strategy that grants to preserve a prescribed size distribution even when large particles are rejected in the growing process.

Recarey et al. (RECAREY et al., 2019) proposes a new approach to particle packing using advancing front to fill a domain while respecting a probabilistic size distribution. Although the technique generates a dense medium, the porosity can be after readjusted

removing a certain number of particles to meet the desired filling ratio.

An important aspect of geometric techniques is the reduction of computational time when compared to other classes of methods, such as dynamic algorithms. The use of dynamic algorithms to modelling huge problems can be computationally expensive and take several hours. Possible solutions to improve computing time are presented by Baugh and Konduri(BAUGH; KONDURI, 2001), and Cintra et al. (CINTRA et al., 2016a). In their work, they show that it is possible to reduce the convergence time of large DEM simulations using High Performance Computing (HPC).

On the other hand, traditional geometric algorithms have difficulty in obtaining responses that meet the specified particle sizes and dynamic equilibrium. In certain types of problems, to overcome theses issues, geometric techniques can be used, as described by Frery et al. (FRERY et al., 2011). This work seeks to meet both the fill rate and size distribution of real granular models with spatial homogeneity, obtaining results close to the experimental data of fully saturated soils.

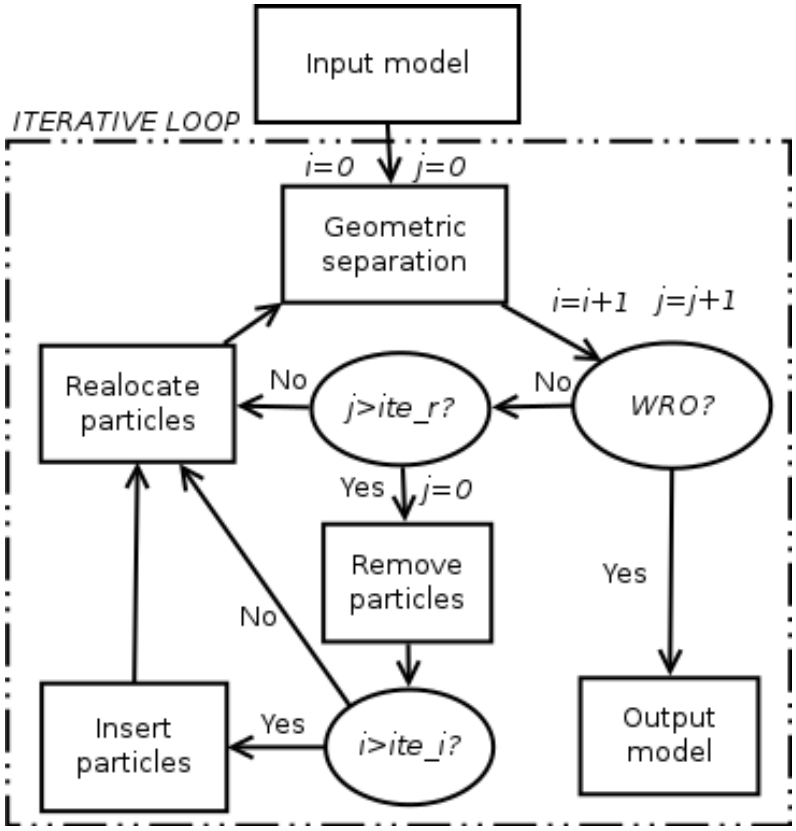
Based on the above explanations, the present work proposes an alternative method of particle packing, with a non-uniform size distribution and a prescribed filling rate on arbitrary domains, using only geometric procedures. The proposed method aims to reproduce characteristics of real granular models, such as soils. The adopted methodology also ensures homogeneity for the spatial distribution of the particle set. Due to the set of constraints to meet the characteristics of desired granular models, additional treatments are realized to reduce computation time and obtain output models close to real particle aggregates. The experimental data presented by Frery (FRERY et al., 2011) are used to validate the results of the proposed method, and practical examples present in the literature are reproduced to illustrate its applicability.

3.2 PROPOSED STRATEGY

The geometric separation method for particle packing proposed in this work is an iterative approach applied to problems where it is desired to prescribe the radius of the particles. It is a simple process, easy to implement in parallel, and involving only geometric parameters. This method initially generates random positions for the particles on domain and, using an iterative procedure, it minimizes the existing overlaps between them by applying incremental displacements. A scheme of the adopted strategy is illustrated in Figure 14.

As described in the Figure, this strategy aims to achieve a Well Reduced Overlapping (WRO) state. In this state, both particle-particle and particle-contour overlapping are satisfactory reduced. The input model is defined using target parameters of the model, such as filling ratio and particle size distribution. The number of particles and their

Figure 14 – Adopted strategy

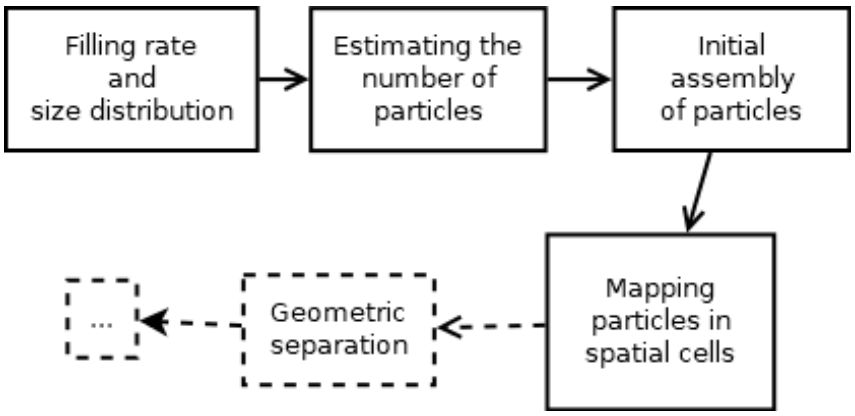


random initial positions are set only once in this step. The following steps are repeated in loop until reaching the WRO state. These other steps of the procedure are strategically used and repeated to reduce these overlaps: geometric separation, particle reallocation, particle removal and particle insertion. The geometric separation moves the particles that overlapped. If the WRO state is not achieved after the geometric separation, the particle reallocation or the particle removing are performed. The reallocation moves overlapped particles from most filled areas to empty areas, improving the homogeneity of the model. If the density of overlapping particles are not reduced to a satisfactory proportion after several geometric separation and reallocation combined attempts, some particles are removed from the domain. In the figure, the index j computes the number of iteractive steps for the particle removal. If the counter j exceeds ite_r, it is necessary to remove particles. After a few steps, however, the removed particles are reinserted into the empty areas of the domain. The index i indicates a counter for the total number of geometric separation iteractions. If a limit of particle reinsertion is reached and the i counter exceeds ite_i (the maximum number of iteractions per inserted particle), then the removed particles stay outside the model. This permanent removal is not desired because it affects the output configuration of the filling ratio and the size distribution of the particles.

3.2.1 Input model definition

The data for the input model of particles are the filling ratio of the model and the grain size distribution. The workflow of this step is shown in Figure 15. The initial number of particles is estimated from this information, as described in Section 3.2.1.2. Those particles are inserted one by one on the domain at random positions.

Figure 15 – input model definition



All generated particles are mapped in spatial cells using the NBS algorithm (MUNJIZA; ANDREWS, 1998). This subdivision of domain by cells helps to search for overlaps using the particle neighborhood instead of the full domain.

3.2.1.1 Filling rate and size distribution

Filling ratio and size distribution are important data related to granular media. A numeric value can be described as shown in the Eq. (3.1) , where S_R represents the filling ratio, also known as solid ratio:

$$S_R = \frac{V_s}{V}, \tag{3.1}$$

where V is the volume of the domain, and V_s is the solid volume. In soils, this filling rate generally have an explicit relationship with porosity. It can be computed as the ratio of the empty volume (not filled by grains) and the total volume, given by:

$$\eta = \frac{V_v}{V}, \tag{3.2}$$

where V_v is the empty volume, and η represents the porosity. As previously cited, this work focuses on two-dimensional problems. Therefore, the solid and empty volumes must be represented by the equivalent areas.In the case of circular particles, this conversion is made treating each particle as a cylinder of unitary height. The equivalence of volumes to areas follows according to the expression:

$$A^* = \frac{V}{h}, \tag{3.3}$$

where h is the unitary height, and A^* is the equivalent area related to the volume V. The filling ratio S_R and the porosity η in a domain are related as described in Eq. (3.4).

$$S_R = 1 - \eta. \tag{3.4}$$

A size distribution for a collection of grains could be defined by discrete ranges of particle sizes, or grain size fractions. It is possible to estimate the total number of particles of a model combining both size distribution and filling ratio according to the equation:

$$(1 - \eta)V \approx \sum_{i=1}^{g} w_i(1 - \eta)V, \tag{3.5}$$

where w_i represents the occupation fraction for the i-th grain size fraction, V is the representative volume of the domain, g indicates the number of fractions and $(1 - \eta)$ is the model's filling ratio. w_i is the area ratio of each fraction, its value must be positive and need to meet:

$$\sum_{i=1}^{g} w_i = 1. \tag{3.6}$$

The combination of the w_i values and the corresponding set of diameters d_i is called grain size distribution. the number of particles and volume that corresponds to each size fraction respects the equation:

$$\sum_{i=1}^{g} n_i v_i = \sum_{i=1}^{g} w_i(1 - \eta)V, \tag{3.7}$$

in which v_i is the correspondent volume for the size fraction i, and n_i is the total number of particles that are part of this fraction.

3.2.1.2 Estimating the number of particles

The total number of particles can be estimated using the grain size distribution and porosity values as input data, and using the Eq. (3.7). Despite the numerical value of the filling ratio being in a continuous range, the total number of particles n is an integer number. So, the value of n calculated in the equations 3.8 and 3.9 must be truncated. An inherited numerical error is a natural consequence of the discontinuous nature of the granular model and the truncated value. This error is minimized when increasing the number of particles. The general equation shown in Eq. (3.7) can be modified to be applied in different cases of interest. For instance, the Eq. (3.8) represents the same relation applied to a model that uses cylinders with diameter d_i and hypothetical unitary height:

$$n = \sum_{i=1}^{g} \frac{4(1 - \eta)V w_i}{\pi {d_i}^2}, \tag{3.8}$$

or, in the case of each single grain size fraction:

$$n_i = \frac{4(1-\eta)Vw_i}{\pi {d_i}^2}. \tag{3.9}$$

For the general case of spherical particles, the number of particles can be estimated in a similar way:

$$n = \sum_{i=1}^{g} \frac{8w_i(1-\eta)V}{\pi {d_i}^3}. \tag{3.10}$$

The inherited error in the calculus of the total number of particles n is also present in the calculus of the number of particles for each grain size fraction n_i. This error could be expressed according to:

$$n_i = floor(\frac{(1-\eta)Vw_i}{v_i}), \tag{3.11}$$

where (for disk shaped particles),

$$v_i = \frac{\pi {d_i}^2}{4}, \tag{3.12}$$

or (for sphere shaped particles),

$$v_i = \frac{\pi {d_i}^3}{8}, \tag{3.13}$$

Equations 3.12 and 3.13 show different ways to calculate v_i for each particular case of particle shape. This calculus can change depending of the chosen shape for the particles. In this equations $floor$ is a mathematical operator that rounds a number down to the nearest integer. It is possible to measure the error of the entire set in terms of the filling ratio residue. This value can be calculated as a function of the filling ratio S_R:

$$\epsilon = 1 - \left(\frac{V - \sum_{i=1}^{g} n_i v_i}{(1-S_R)V} \right). \tag{3.14}$$

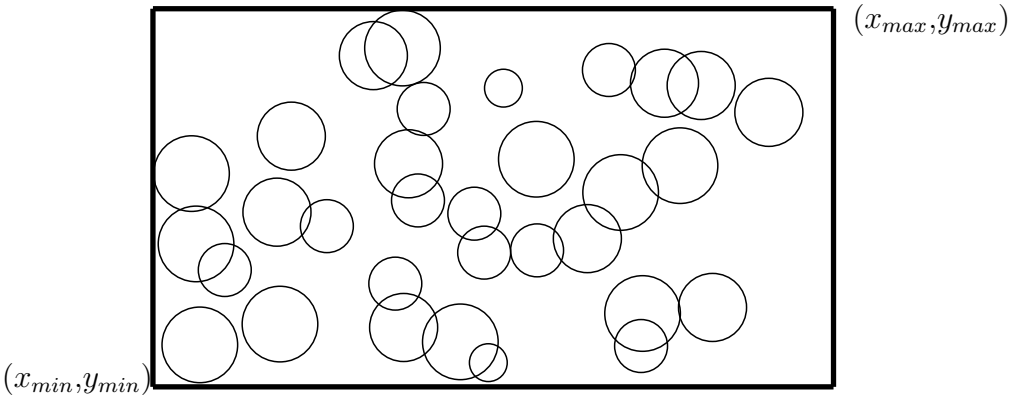
Tests show that models composed with particles of great size in comparison with the full domain also have higher filling ratio residues, while models with smaller particles present smaller values of error. Therefore, the domain discretization using smaller particles tends to reduce the error values.

Despite mathematical representation of the maximum quantities of particles to be inserted, nothing was considered about the geometric shapes and the particle spatial distributions yet. As the input model is obtained by inserting the particles in random position, and it does not consider avoiding overlapping, the geometric shape and positions is only considered in the following steps.

3.2.1.3 Particle distribution in the domain

This step consists in a particle's random distribution on the domain from the quantities estimated in the subsection 3.2.1.2. Figure 16 illustrates an initial particle assembly.

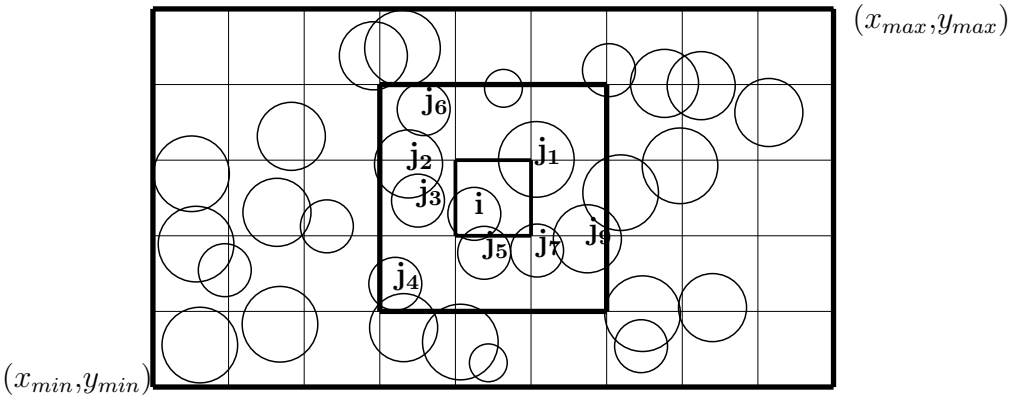
Figure 16 – Initial particle assembly in a bi-dimensional example, showing overlapping between particles.



The particle coordinates are randomly set on the domain one by one, ensuring the radii proportion and quantities. As there are no constraints in this distribution, the particles can overlap each other. The strategy aims to mitigate these overlapping along the iteractions.

3.2.1.4 Mapping particles into spatial cells

Figure 17 – Domain divided by cells. The external bold square contains the j particles in the immediate neighborhood of the particle i, and the internal bold square is the cell that contains the particle i.



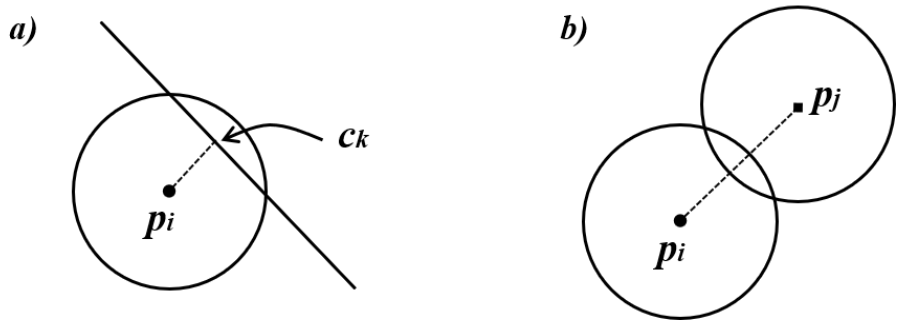
All particles are mapped into spatial cells that represent domain decomposition. This mapping is important for the efficiency of the algorithm and a prerequisite for the geometric separation procedure, as described in the section 3.2.3. Particles can overlap each other and the domain limits (or contours). Verification of particle-by-particle overlay may require a high computational cost as the number of particles increases. Cell mapping algorithms allow you to divide particles into nearby neighborhoods to reduce the search area for overlaps within those neighborhoods rather than the full domain. Figure 17 illustrates the cell division. The bold smaller square is the cell that contains the particle i, and the larger bold square represents the immediate neighborhood that contains the particles j_k. In conclusion, searching for particles and incidences in near neighborhoods is computationally more efficient than searching on full domain.

The literature presents techniques of cell mapping often adopted in Discrete Element Method simulations (MUNJIZA; ANDREWS, 1998; WILLIAMS; PERKINS; COOK, 2004). This work uses the classic algorithm NBS used by Munjiza and Andrews (MUNJIZA; ANDREWS, 1998), where each cell is a square space with edges measuring two times the larger radii size of all particles. In general, searching algorithms are conceived and studied to this default cell size. Cells with other sizes may not work properly, presenting nonconformity with the standard setting.

3.2.2 Overlapping mapping

An important step for the geometric separation procedure is mapping overlaps and contacts. Mapping of overlapping particles is performed within each cell. A contact occurs when two particles overlap, *ie*, the center distances of these particles is lower than the sum of their radii. Contacts can also occur when particles overlap to domain boundary. These two types of contact iteractions are described in Figure 32. Contact mapping is also used in the particles removal and reallocation. Only particles that overlap each other need to be relocated and/or removed.

Figure 18 – Domain cell decomposition and cell mapping.



3.2.3 Geometric separation

Geometric separation is the main method of changing particle positions in the proposed strategy. This process aims to eliminate the particle overlaps. Through an iterative procedure, the approach minimizes overlaps by applying incremental displacements. The position of an arbitrary i particle is defined as:

$$\boldsymbol{p_i} = \{x_i, y_i\}, \tag{3.15}$$

x_i and y_i are the Cartesian coordinates. The evolution of these coordinates over iterations is computed by:

$$\boldsymbol{p}_i^{m+1} = \boldsymbol{p}_i^m + \boldsymbol{\delta}_i, \tag{3.16}$$

where m indicates the iterative step index and δ_i is the corresponding calculated displacement. The calculation of displacements considers only contact particle pairs from a set

of neighboring elements n_p. The corresponding geometric overlap ϵ_{ij} and displacement increments are calculated for each pair of contacts $[i, j]$. These increments are computed according to the the contact plane normal vector and the radii of the particles $p_i - p_j$, r_i and r_j, in the form of the following equations:

$$\boldsymbol{\epsilon}_{ij} = r_i r_j - ||p_i - p_j||, \tag{3.17}$$

$$\boldsymbol{\delta p}_i = \sum_{j=1, j\neq i}^{n_p} \Phi_{ij} \frac{\epsilon_{ij}}{||p_i - p_j||}(p_i - p_j), \tag{3.18}$$

$$\boldsymbol{\delta c}_i = \sum_{k=1}^{n_o} \Phi_{ik} \frac{\epsilon_{ik}}{||p_i - c_k||}(p_i - c_k), \tag{3.19}$$

$$\boldsymbol{\delta}_i = \boldsymbol{\delta p}_i + \boldsymbol{\delta c}_i. \tag{3.20}$$

The the sub-index p belongs to the variables referent to particles, and the sub-index c to the variables referent to obstacles. ϵ_{ik} is a numerical value to measure the overlap of the particles and the domain boundary (or interior obstacles). ϕ_{ik} and ϕ_{ij} are matrix elements that can assume unit values to contacts between the corresponding i, j elements, and zero value otherwise. Its value could be expressed by:

$$\boldsymbol{\epsilon}_{ik} = r_i - ||c_k - p_i||, \tag{3.21}$$

c_k are the coordinates of the contact point, as described in Figure 18. The cells containing each obstacle are computed, and the contact between the obstacle and the particles in these cells are mapped. The iterative procedure occur until the maximum overlap is lower than a tolerance value. The convergence rate of the algorithm is, in general, higher for cases in which the average porosity is high. The tolerance value for the overlap can be defined as a proportion of the particle radii. The particle positions are not updated prior to the calculation of the displacements of all the particles.

3.2.4 Reallocating particles

An important step to reduce the computational time required to eliminate overlaps is the reallocation of particles. Particles located in most occupied cells must be reallocated to regions with low filling rate values. This is achieved by using the cell structure proposed in Sec. 3.2.1.4. For each cell, it is computed a numerical value associated to filled area as follows:

$$\boldsymbol{S}_c = \sum_{j=0}^{npc} \pi * r_j^2 + \frac{S_n}{2}, \tag{3.22}$$

where S_c represents the fill area associated with the cell, npc is the number of particles in that cell, r_j is the radius of the particle j inside the cell. The equation considers the

filled area of the cells in the immediate vicinity as S_n. S_n is the sum of the area of all the particles in the immediate vicinity of the cell divided by the number of neighboring cells. Particles inside the most filled cells are reallocated to the most empty ones. Only particles with radius smaller than the average radius are candidates for reallocation, avoiding to create large gaps in the process.

3.2.5 Removing and inserting particles

After executing the reallocation procedure, if the density of overlapping particles are not reduced to a satisfactory proportion after several geometric separation and reallocation combined attempts, some particles are removed from the domain. Particle removal is performed for particles that overlap the most. The number of particles to be removed is related to a numerical value called *porosity increment* (Δ_η). This value should be small (between 0.01% and 1%) to ensure that the output filling ratio minimizes modifications in the particles initial distribution. Thus, fraction of particles area should be removed according this increase in the filling rate:

$$S_R = S_T \Delta_\eta. \tag{3.23}$$

The area of removed particles must be less than area S_R, a portion of the total area S_R. The output filling ratio should also be updated according to the number of particles removed. This process will be repeated simultaneously with relocation until the reduction of overlaps reaches a desired tolerance. The removal of a particle damages compliance of the target parameters defined by the initial model, because the total number of particles to achieve this goal is generated at input model. Repeating the particle removal successive times can generate an elevated filling ratio error, even when respecting the *porosity increment* in each particle removing. In order to avoid great filling rate errors, the particles that are removed in the previous steps can be reinserted in regions with lower filling rate. Inserting the particles back on model reduces the error measure in the initial and output models main parameters (filling rate and size distribution). However, the particle insertion generates new contacts and overlaps, hindering the geometric separation process. This requires other steps to achieve the output model. To avoid this, a limit is settled for attempting to reinsert particles. If the number of iteractions exceeds this limit, then the removed particles are not reinserted.

3.2.6 Influence of the graphic standard deviation

In order to reduce the filling error values described in section 3.2.5, a strategy using a response surface is proposed. Some considerations for grain diameters are generally made when representing soils by size fractions and distributions. A common representation is the Krumbein Phi scale (KRUMBEIN, 1937). The grain diameters are measured on this

scale and expressed by the equation:

$$\Phi = -\log_2(\frac{D}{D_o}), \tag{3.24}$$

where D represents the grain diameter and D_o represents a reference diameter, usually equal to the working unit.

A scalar number called the inclusive graphic standard deviation σ_1 (UDDEN, 1914) is used to express and qualifying soil distributions using this scale. The inclusive graphical standard deviation is a value that measures the homogeneity of the grain size distribution, and is expressed in Eq. 3.25:

$$\sigma_1 = 0.25(\Phi 84 - \Phi 16) + 0.15(\Phi 95 - \Phi 5). \tag{3.25}$$

Frery et al. presented seven examples of granular models generated from different soils. The data provided from a sieve analysis is represented by probabilistic log-normal distributions, as shown in Table 9.

Table 9 – Probabilistic normal distribution of soil samples (FRERY et al., 2011)

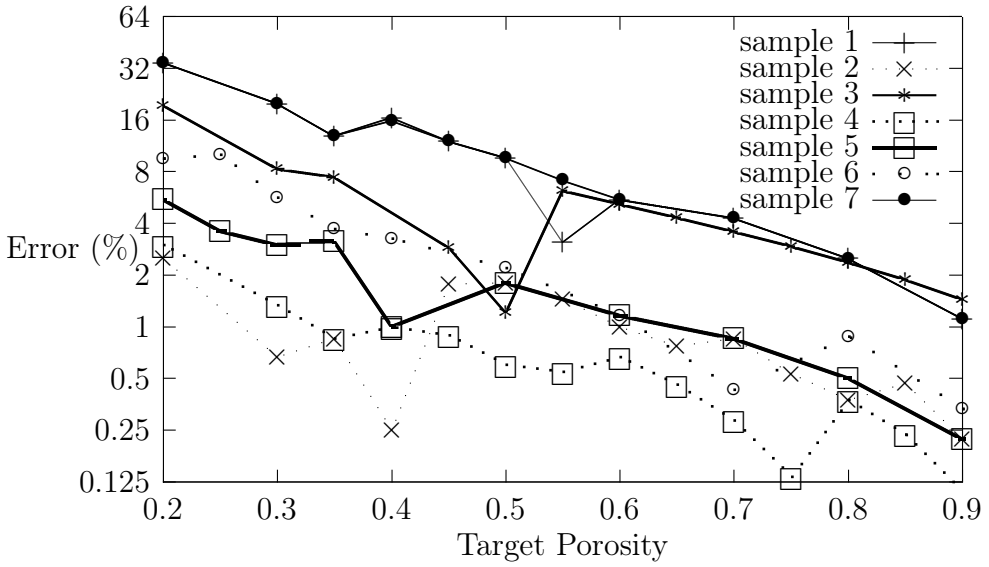
Sample	μ	σ
1	2.338	0.159
2	2.366	0.054
3	2.393	0.113
4	2.354	0.051
5	2.274	0.095
6	2.289	0.106
7	2.403	0.055

Table 10 – graphic standard deviation of soil samples (FRERY et al., 2011)

Sample	$S.Deviation$	$Classification$
1	2.1591	very poorly sorted
2	0.9545	moderately sorted
3	1.7576	poorly sorted
4	0.9545	moderately sorted
5	1.3560	poorly sorted
6	1.3560	poorly sorted
7	2.1591	very poorly sorted

The samples data are used to generate granular models and to analyze the influence between the standard graphic deviation in the error when comparing the target filling index with that obtained in the output disk packing. The value of porosity is used to characterize the models instead of the filling ratio because the data is related to soils.

Figure 19 – Porosity error using the size distribution of the samples.



Several numerical models are generated based on the grain size distributions of the samples, varying only the porosity. The error values when comparing the input and output model's porosity is shown in is shown in Figure 19.

Samples presented by Frery are composed by silt, clay and sand, that comprehend mostly values of Φ from 0ϕ to 8ϕ . For these examples, the standard graphic deviation is described by Table 10.

Samples close to the very poorly graded classification (samples 1 and 7) show the highest values of error. There are two proposed solutions to overcome the modeling limitation for these soil classes: a) increase the number of particle reinsertion attempts and b) predict the filling rate error by using a response surface. Increasing the number of reinsertion attempts is a way to try avoiding unnecessary particle removal. However, this approach has shown a high computational cost and sometimes it is impossible to meet the desired parameters.

Using the examples provided by Frery et al. (FRERY et al., 2011), a response surface is created to approximate the output model and the input model, mitigating the values of filling rate error. Frery et al. (2011) presented seven examples of granular models generated from different soils. The data provided by a sieve analysis is represented by probabilistic log-normal distributions, as shown in Table 9. The error values in Figure 19 are used to adjust the response surface. Eq.3.26 represents the polynomial equation that correlates the input porosity to the expected output value and also represents the response surface. The values of the coefficients A_i, B_i, C_i, and D_i are presented in the

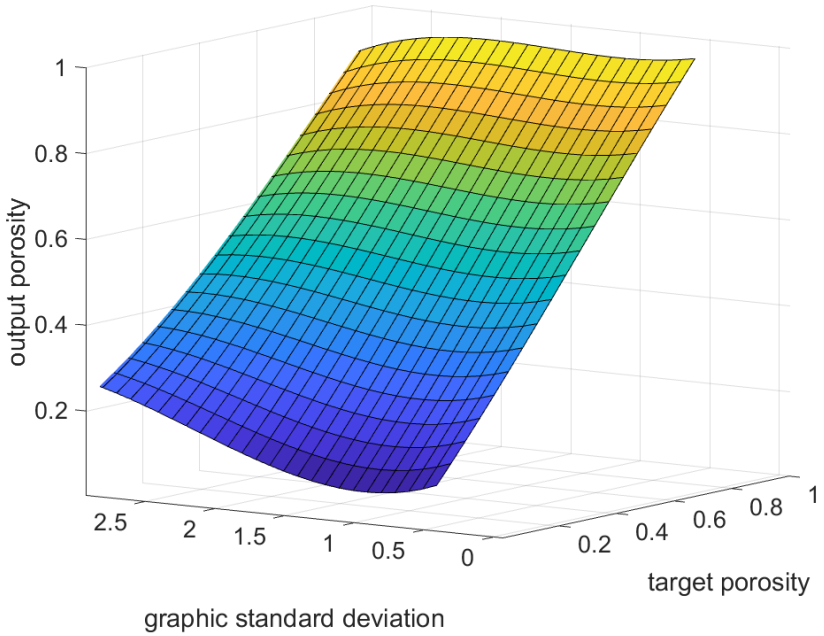
Table 11, and the function visual aspect is presented in Figure 20.

$$\eta_f = \sum_{a=0}^{3} A_a \eta^a + \sum_{b=0}^{3} B_b \sigma_1^b + \sum_{i=0}^{2} C_i \eta^i . \sum_{j=0}^{2} D_j \sigma_1^j \tag{3.26}$$

Table 11 – Response surface coefficient values.

	0	1	2	3
A	0.34142095	0.16536358	0.55049502	-0.16158216
B	-1.38865518	-0.80876701	0.44287278	-0.02837543
C	-0.73466345	-0.35128767	0.14242163	
D	-1.65576392	-0.6725995	0.3340948	

Figure 20 – Response surface representation.



η_f represents the output porosity obtained by the response surface. It is possible to calculate a η value that makes η_f equal to that targeted in the input model. A intermediate model is then created using this calculated η, called response surface model (RM model). The steps described in Figure 15 is modified to include this new step, as described in Figure 21.

The following results are the porosity error for models generated using the size distributions from the seven samples and varying the porosity values using the response surface method and Response Surface (RS) models. Figure 22 shows the relationship between the target porosity and the error (%) in this approach. The results show that error values are mitigated below 4% for all samples if the input porosity is greater than 30%, and only

Figure 21 – Response surface method

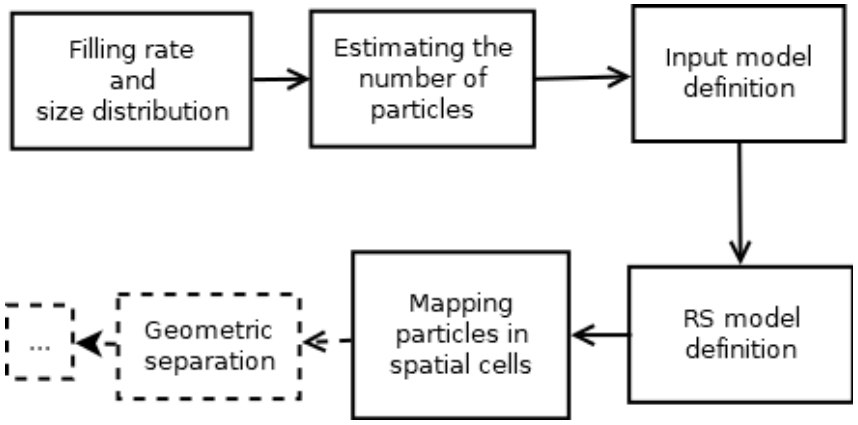
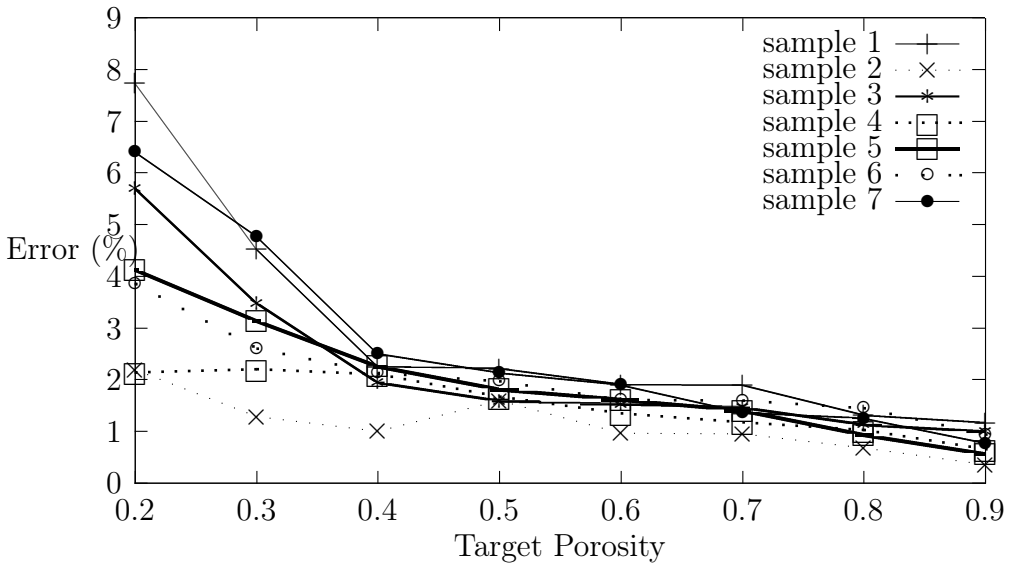


Figure 22 – Porosity error using the response surface.



samples classified as very poorly graded have large porosity errors when using smaller input values. So, the response surface method is included in the strategy in order to obtain more accurate output models.

3.3 Numerical studies

This section discusses the parameters employed in the results, as well as the solutions presented to overcome certain limitations presented by the proposed method. The results are detailed in the following subsections. The strategy is validated by comparing the results with the numerical and experimental data presented by Frery et al. (FRERY et al., 2011), whose focus is on soil representation.

3.3.1 Strategy validation

As stated in 3.2.1.2, the input data for the models are the porosity values and grain size distribution in the numerical study conduced in this section the samples presented by Frery et al (2011), as described in the Table 12, are used.

Table 12 – Porosity values of soil samples.

Amostra	η
1	0.350
2	0.430
3	0.420
4	0.390
5	0.400
6	0.330
7	0.440

Table 13 – Results for porosity values of soil samples and numerical models (FRERY et al., 2011).

Sample	η	η_{Frery}	η_L	η_{SG}
1	0.35	0.345	0.346	0.353
2	0.430	0.421	0.427	0.430
3	0.420	0.418	0.422	0.431
4	0.390	0.390	0.381	0.395
5	0.400	0.399	0.392	0.403
6	0.330	0.328	0.340	0.339
7	0.440	0.430	0.448	0.439

The objective is to evaluate the error in porosity and grain size distribution for each grain size fraction. The seven soil sample distributions presented in Frery's work (FRERY et al., 2011) are reproduced as granular models for the validation of this proposed geometric separation strategy. All samples are generated in square domains with a number of particles varying between 15,000 and 100,000, according to model's grain size distribution and porosity. The obtained results and the particle size distribution profiles obtained with the proposed strategy are presented in Tables 13 and 14. In the Table 13, η represents the porosity value of the experimental data, the third column of the table corresponds to the results presented in the work of Frery, the fourth column contains the results obtained by the authors in a previous work using the distance minimization approach, and the last column shows the results obtained with the strategy proposed in this work.

The proposed method also attempts to achieve a target size distribution. This means that the proportion between the different radii sizes in the output model must be equal or at least close to the input data (that is not necessarily equal to the input model due the response surface). An investigation of this aspect is made by studying the percentage

difference between the number of particles, in each fraction of grain size,considering the input data and the output model. The results are described in Table 14.

Table 14 – Error by grain size fraction for each sample

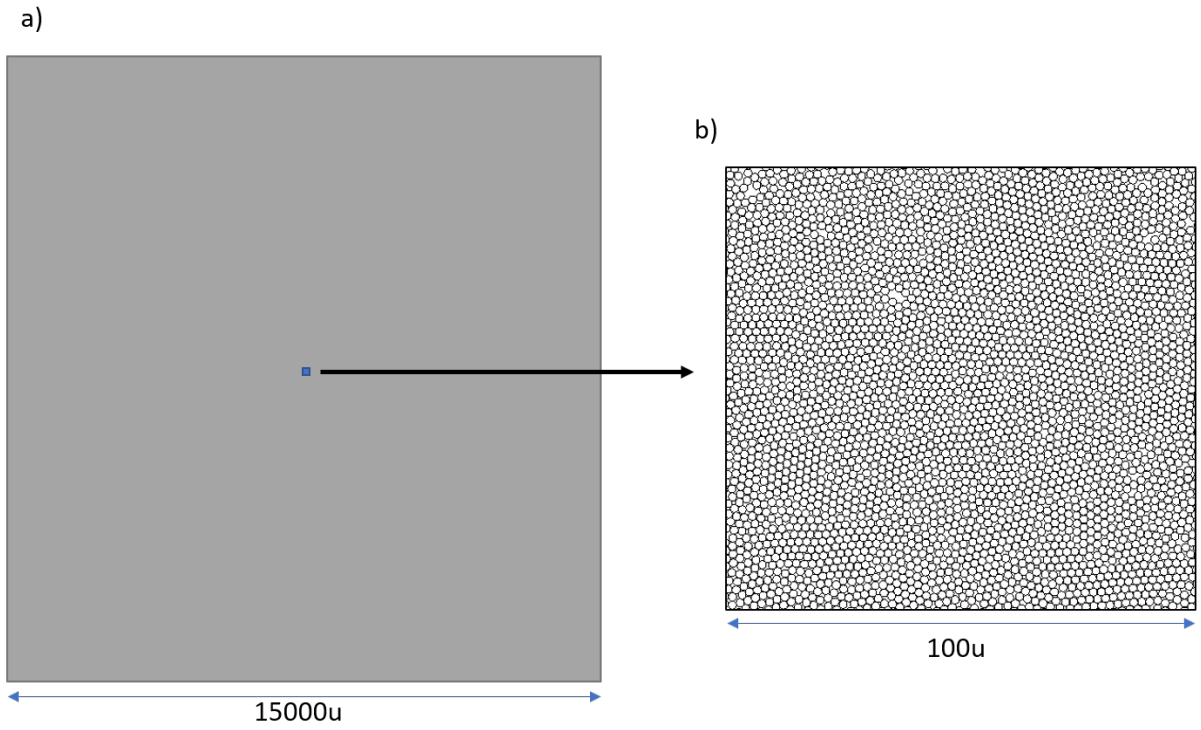
Sample	0ϕ	1ϕ	2ϕ	3ϕ	4ϕ	5ϕ	6ϕ	7ϕ	8ϕ
1	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	1.288%	3.294%	2.0%
2	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
3	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	1.0%	1.2%	3.5%
4	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
5	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
6	0.01%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
7	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	1.3%	0.988%	0.8%

The maximum error value shown in the Table 14 is less than 5%, and most samples exactly respect the target size distribution. These results show that the target size distribution is closely respected in the output model.

3.3.2 Examples

Granular models are generated in this section to illustrate the applicability of the strategy.

Figure 23 – Model with 2,291,829 particles. a) Dimension of the entire square domain and b) A representative sample.



The first example is granular model represented for a single fraction generated in a square box domain. The target porosity of the first example is 20% and the model has a grain size distribution shown in table 15 . The input model of the model has 2,291,882 particles, and the output model has 2,291,829 particles, as shown in Figure 23. The output porosity achieved is 20.00002%, a close value despite the number of particles removed. The Table 15 shows the data profile for the generated example, in which both the size distribution and the target porosity are achieved within the expected tolerance.

Table 15 – Data profile of the example 1.

Radius	**Target**	**Output**
1.00	100.00 %	100.00 %
Porosity	**20%**	**20.00002%**
Particle Number	**2,291,882**	**2,291,829**

Figure 24 – a) Example 2: granular model generated in a circular domain; b) Example 3: circular mill.

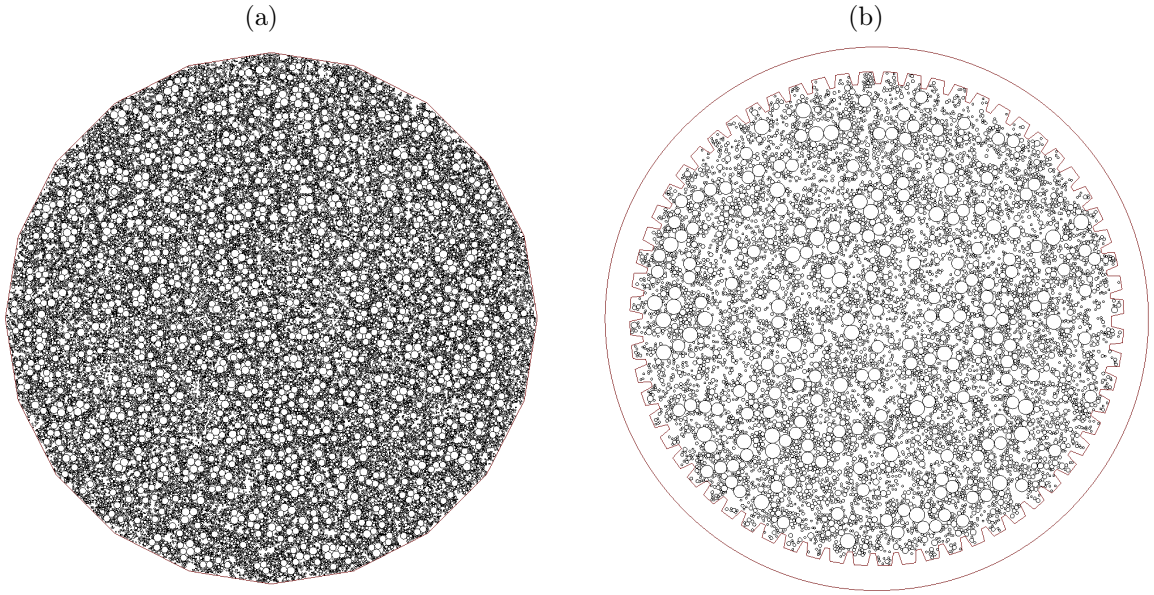


Table 16 – Data profile of the examples 2 and 3

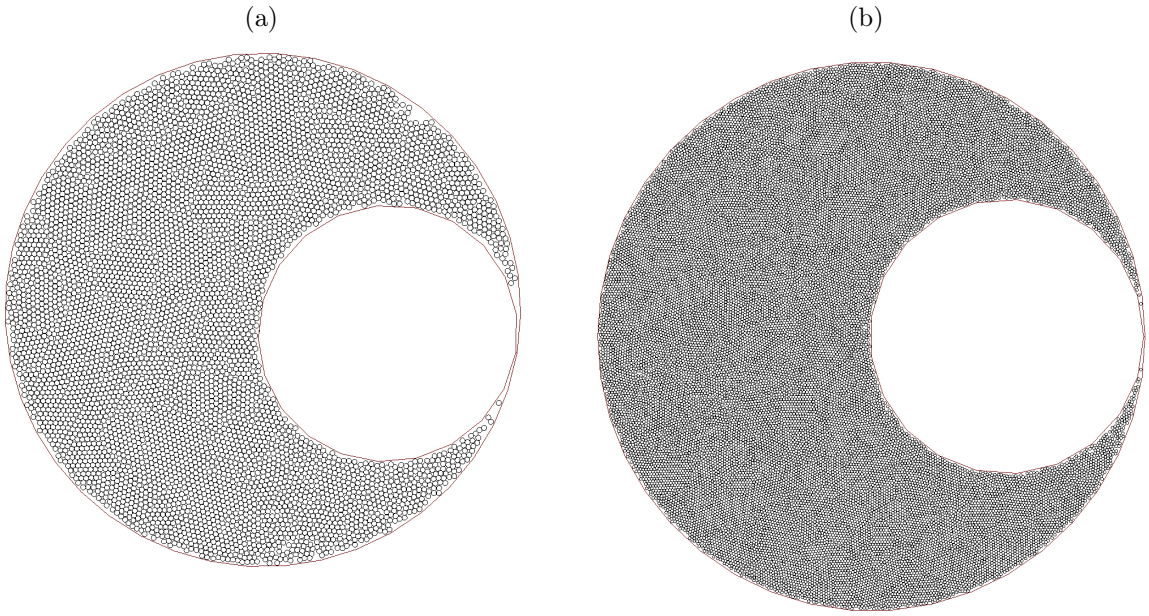
Radius	**Target**	**Output a)**	**Output b)**
0.02	22.00%	22.17%	22.19%
0.03	16.00%	16.12%	16.12%
0.04	13.00%	13.08%	13.11%
0.06	13.00%	13.08%	13.00%
0.125	26.00%	26.09%	25.94%
0.15	10.00%	9.45%	9.67%
Porosity	**a) 40%/b) 50%**	**40.14%**	**50.19%**

The methodology is also applicable to generations within non-square domains, and it is possible to fill several arbitrary polygons, as shown by example 2 in Figure 24. The Figure shows in the example 2 a circular domain with a 10 units radius, being represented by a polygon with 20 edges. Table 16 shows grain size profiles of this example.

It is observed that the number of edges to refine the contour of the circle will impact the computation time. The section 3.4 describes the relationship between the number of edges and the time required to generate a model.

Example 3 is a mill represented by a non-convex polygon with 320 edges, as it is also shown in Figure 24. The mill model is based on the example presented by Xu et al. (XU; LUO; ZHAO, 2018). Table 16 shows the details of this example.

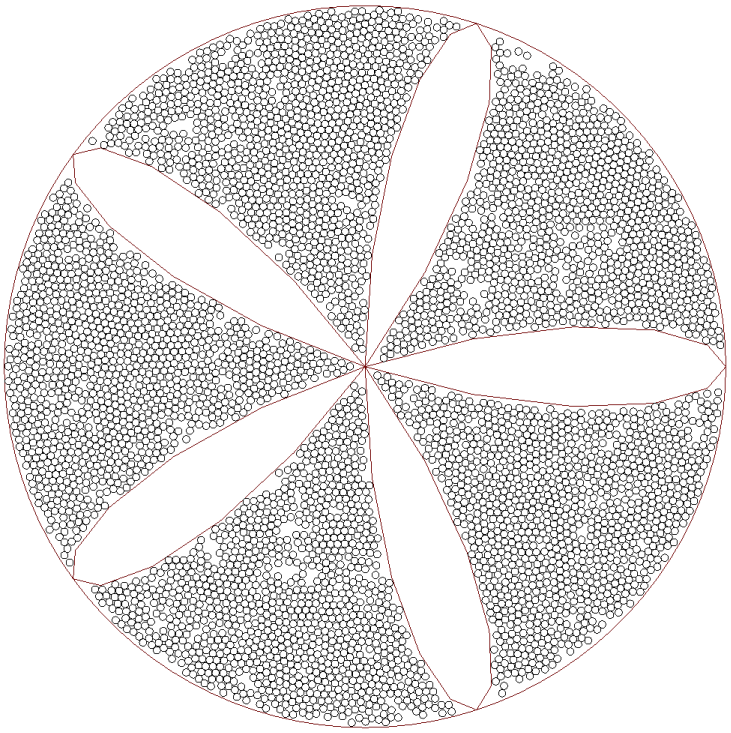
Figure 25 – a) Example 4: model with cylinder obstacle; b) Example 5: model with cylinder obstacle and small particles.



The examples 4 and 5, shown in the Figure 25, illustrate applications with internal obstacles, and are based on the models presented by He et al. (HE; BAYLY; HASSANPOUR, 2018). The models are composed by particles, with a radius equal to 10 units, containing a large object represented by a inner circle with half the diameter of the other circle. It is possible to observe in the example 4 that the packing algorithm can present empty gaps near the contours. This happens when the particle size is in the same order of magnitude as areas to be filled. Example 5 shows a model with ten times smaller particles,in which the voids near the contours are reduced.

Shan and Zhu (SHAN; ZHU, 2018) present a model for predicting dynamic cutting forces when milling sand molds. Example 6 (Figure 26) shows a similar propeller mill without cutting edges, illustrating the capabilities of the packaging method in these cases.

Figure 26 – Example 6: engine mill.



The presented models in this section show that the packing strategy can be used in domains with varied geometries, respecting both the target filling ratio and the size distribution.

3.4 Computational performance

The studies presented in this section are related to the computing time demanded by the main tasks performed by the methodology, and they show that some tasks take longer to complete than others. The most important tasks in terms of computing time are divided into: a) geometric separation solver that computes the overlapping of particles; b) cell mapping; c) execution of sorting algorithms during the particle reallocation, removing and inserting particles procedures. The reallocation, removal and reinsertion steps are very similar procedures, and most of the time required for them (above 98%) is due to the execution of sorting algorithms. The Table 17 shows the method's response by varying the ratio between the larger and smaller particle radii of the models.

Each column that relate to a task shows the fraction of the total time demanded to execute that task, and the second column shows a ration of time demanded per particle. All the examples are executed in an Intel core i5 CPU, with four cores at 3.2 GHz and 16 GB of Ram memory. The results show that the ratio between the maximum and minimum radii directly influences the calculation time per particle. Regarding the iteration steps, the greatest contribution to the computation time comes from the geometric separation

Table 17 – Time profile varying the relation between the the larger and smaller radii.

R_{max}/R_{min}	**Time/particle**	**Solver**	**Cell's mapping**	**Sorting***
1	0.0126 s/part	56.63 %	4.36 %	39.00 %
2	0.0495 s/part	57.1 %	1.61 %	41.342 %
4	0.0712 s/part	93.22 %	1.22 %	5.552 %
8	0.0844 s/part	98.86 %	0.38 %	0.75 %
16	0.0985 s/part	99.63 %	0.097 %	0.26 %
32	0.105 s/part	99.69 %	0.072 %	0.238 %
64	0.112 s/part	99.75 %	0.05 %	0.19 %
128	0.450 s/part	99.91 %	0.047 %	0.05 %
256	1.068 s/part	99.96 %	0.01 %	0.03 %

solver. This solver performs the search of particles in the neighborhood along with the calculation of displacements. The percentage corresponding to this process is also affected by the ratio between the largest and smallest radii in the same way that the calculation time per particle is affected. This is a direct consequence of the grid table used in the neighborhood search, because the cells have a constant size in relation to the largest radius.

Another important factor that influences computing time is the number of obstacles and contour edges in the model. table 18 shows the correspondence between the number of edges and the time required to generate the example described in Figure 24. The results

Table 18 – Time results varying the number of edges

Number of edges	**Time (s)**
10	5.2
100	17.5
200	60.5
300	198.2
400	318.7
500	478.8
600	671.8
700	739.8
800	1094.9
900	1379.5
1000	1768

show that the increase in the number of edges and obstacles of the domain also directly impacts the computing time required to generate the models. This happens because mapping contour overlapping requires more computing time when increasing the number of obstacles that belongs to the domain boundary. Parallel implementation may benefit the computing time for those cases, because the calculus of the contour and obstacle overlapping are independent tasks.

3.5 Commentaries about the technique

This chapter presented a particle packing algorithm with non-uniform sizes and prescribed filling rate. The output granular model is obtained by geometric separation of the overlapping particles that are initially generated in random positions. A response surface is also proposed to minimize the error associated with the input filling rate and that obtained in the output model. This allows better results without compromising the computing time.

The strategy is validated by comparing the results with experimental and numerical data available in the literature. Examples are also presented to illustrate the applicability of the strategy in domains of varied geometry.

A natural evolution of the algorithm involves a three-dimensional version of the strategy. However, studies are still needed to obtain a better understanding of the correlation of the parameters used for three-dimensional models.

This strategy achieves the main objective of generating particle arrangements with a prescribed filling ratio and particle size distribution for arbitrary domain geometries. The proposed method presents good accuracy in the reproduction of the input data, even with smaller values of porosity and great variations in the grain size distribution. The main limitation of the proposed strategy is the computing time demanded to execute some tasks depending on the input model values. Studies on the computational cost of the generated samples show that parallelization strategies could be used to improve the convergence time (BAUGH; KONDURI, 2001; CINTRA et al., 2016a).

4 A PARALLEL GEOMETRIC PARTICLE PACKING METHOD

This chapter presents a methodology for obtaining granular models from a GPU parallel implementation of the geometric separation particle packing strategy. The text that follows is based on the scientific paper submitted by Lopes, Cintra and Lira (2019b) by the Engineering Computations Journal. This methodology is suitable for the generation of large-scale granular models used in discontinuous media simulations.

The proposed approach uses disk shaped particles (two-dimensional approach) and parallelization mechanisms that consider different computational environments, with a focus on GPUs. The methodology is divided into three macro-steps: a) definition of an input set of particles; b) geometric separation; and c) removal of spurious particles. The set of input particles uses data related to the particle size distribution and the domain filling rate, defining arbitrary positions for the particles. The other steps are used to eliminate overlaps between particles. Parallel computing is performed using the OpenCL programming API on compatible devices.

Examples are presented to show the effectiveness of the proposed methodology. They show good time improvement and better memory efficiency in comparison to the original serial version of the strategy. The method is also validated by comparing the results with experimental and numerical data from the literature.

The proposed methodology allows generating granular models with a parallel GPU particle packing method. It turns possible the achievement of bigger models in smaller amount of time, with little effect in the efficiency and accuracy. It also presents mechanisms to avoid information exchange between GPU and CPU.

The focus of this chapter is to establish and improve performance parameters. Despite presenting examples and the method validation, the main goal is to show a technique that makes good use of the GPU implementations. Because of that, only the changes and improvements of the alternative strategy are explored. The steps that are equal to the original geometric separation methodology are simply cited. This steps are better described in Chapter 3. The text also corresponds as it is presented in the cited article, with format adaptations to the UFAL dissertation criteria.

4.1 Literature review and initial commentaries

The Discrete Element Method (DEM) (CUNDALL; STRACK, 1979) is a well known technique used to achieve significant results from numerica simulations of discontinuous problems involving granular media. Its validation and calibration show good agreement with the behavior of real discontinuous systems in geomechanical and engineering problems

(COETZEE, 2019; DAI et al., 2019), e.g. fragmentation, fracturing, impact and collision phenomena (CAMPBELL; CLEARY; HOPKINS, 1995; BROWN et al., 2000; ONATE; ROJEK, 2004; CHANG; TABOADA, 2009; CLAUSEN et al., 2019).

DEM simulations require an input particle packing that tries to represent closely a real discontinuous media. The numerical granular model is often demanded to attend some requirements: (i) particles with non-uniform sizes; (ii) domain filling with prescribed occupation ratio; (iii) homogeneous distribution of particles; and (iv) global force equilibrium. The literature presents a variety of methods for particle packing, mainly for disk and sphere shaped particles. Even if the characteristics of the problem to be solved may vary, it is always necessary to simultaneously meet these four characteristics or some of them combined. Several approaches have been proposed to obtain a packing of particles that meet the desired parameters.

This work focuses on a category of methods known as geometric packing algorithms, where particles are spatially distributed on domain based on geometric procedures. On the other hand, there are several approaches to compute the spatial distribution of particles from DEM simulations that meet the physical laws and equations of motion. The techniques that use these equations are called dynamic algorithms. Normally, the geometric methods significantly reduces the processing of numerical modeling, since it does not require the solution of the physics equations.

Frery et al. (FRERY et al., 2011) propose the Simple Sequential Inhibition (SSI) process for geometric packing. The method prescribes the positions of particles by the theory of point processes. This technique avoids geometric overlapping of particles, and particle sizes are initially defined by the characteristics of the problem, such an statistical grain size distribution and domain filling ratio. It then tries to place these particles on domain until reaching the desired filling rate. The contributions in Frery et al. work include numerical examples applied to soil media based on experimental data. The particle sizes and the prescribed filling ratio are represented by grain size distribution and porosity. It is generally difficult to accurately represent the size distribution of soil grains, due to the large differences in size between them. The work present soil samples with great differences between the sizes of the grains despite this limitation.

Recarey et al. (RECAREY et al., 2019) present studies on the existing methodologies for particle packing in the literature. Their analysis on the state-of-the-art particle packing generation for DEM makes clear the necessity to continue its development. In this context, they propose an alternative approach to particle packing using advancing front to fill a domain also using a probabilistic size distribution. Although the technique aims to generate a dense media, the porosity can be after readjusted removing a certain number of particles to meet the desired filling ratio. Their methodology presents a low computational cost and intends to meet requirements to engineering applications.

As mentioned by Recarey et al., geometric techniques have the advantage in the computing time when compared to other classes of methods, such as dynamic algorithms. Even so, modeling huge models can be computationally expensive and take several hours depending on the used approach. Baugh and Konduri (BAUGH; KONDURI, 2001), and Cintra et al. (CINTRA et al., 2016a) present methods with possible solutions to improve the computing time demanded in the geometric particle packing. In their work, they show that it is possible to reduce the convergence time of large DEM simulations using High Performance Computing (HPC).

Cintra et al. (CINTRA et al., 2016a) show how some of the physical parameters of numerical granular models require the solution of a large number of interactive equations. The limiting factors for the efficiency of packing algorithms are the number of particles and the number of solution steps. It is common to use a large number of particles to represent more faithfully a given domain in practical DEM applications. The computational models used in particle simulations have been using an ever-increasing amount in the number of particles along years. Large scale problems, such as the one presented by Walther and Sbalzarini (WALTHER; SBALZARINI, 2009), may involve media containing hundreds of millions of particles.

DEM implementations must be executed in computational systems capable of manipulate a great amount of processing and data. Cintra et al. (CINTRA et al., 2016b) show that memory access management can improve significantly the computing time demanded on DEM simulations. In order to mitigate this problem, some processing parallelization techniques have been used, such as those that involve numerical or graphic processors (CPU's and GPU's) on shared or distributed memory environments. Parallel programming resources such as the OpenCL API (GASTER et al., 2013) is currently used for this purpose.

Govender, Wilke and Kok (2016) present a DEM simulation framework called Blaze-DEMGPU that take advantage of GPUs to work with discrete models composed by millions of particles. The high level performance is attributed to the light weight and Single Instruction Multiple Data (SIMD) that the GPU architecture offers. Govender et al. (2015) and Pizette et al. (PIZETTE et al., 2017) present practical engineering applications for DEM simulations using GPUs. These results show the GPUs potential to enable solutions of large-scale problems using discrete granular approaches.

Based on the above explanations, this paper proposes a particle geometric packing technique using parallelization mechanisms with focus on GPUs. The adopted methodology ensures homogeneity for the spatial distribution of the particles and desired model's characteristics, such as the domain filling ratio and particle size distribution. Sorting and domain decomposition methods are developed specifically to allow the method to be entirely executed on a single device (GPU or CPU). This optimize the memory usage, avoiding

information exchange between different devices. Heterogeneous hardware computing is performed through the OpenCL programming API on compatible devices. Examples are presented to validate and show the effectiveness of the strategy, comparing its performance in several devices.

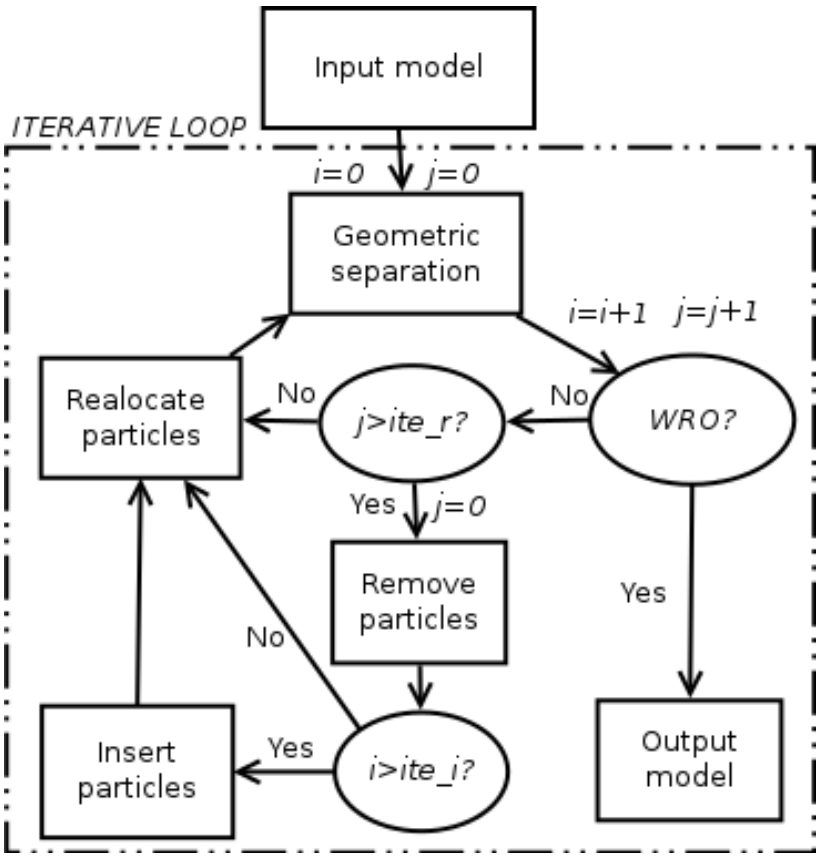
4.2 PROPOSED STRATEGY

The parallel geometric separation method for particle packing using GPU proposed in this work is an iterative approach applied to problems with prescribed particle radii. It is relatively a simple process, developed using GPU parallel computing, where its involves only geometric parameters. In previous works, the authors presented the geometric separation methodology, including performance analysis and validation.

4.2.1 Serial geometric separation methodology

This procedure initially generates random positions for the particles on domain. Using an iterative procedure, it minimizes the particle overlaps by applying incremental displacements. A scheme of the adopted strategy is illustrated in Figure 27.

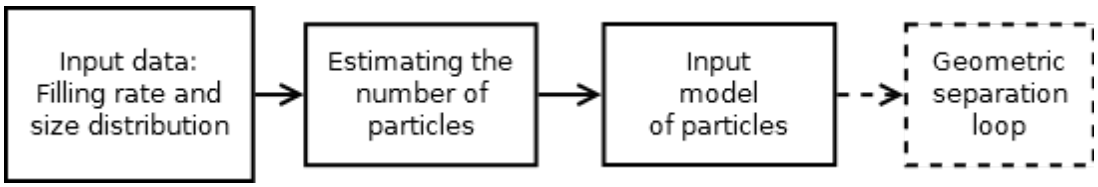
Figure 27 – Geometric separation strategy



The data for the input model of particles are the filling ratio of the model and the grain size distribution. The workflow of this step is shown in Figure 28. The initial number

of particles is estimated from this input information. Those particles are inserted one by one on the domain at random positions.

Figure 28 – Input model definition



As showed in Figure 27, the strategy aims to achieve a Well Reduced Overlapping (WRO) state. In this state, both particle-particle and particle to domain boundary overlapping are satisfactory reduced. The input model is defined using target parameters, such as filling ratio and particle size distribution. The number of particles and their random initial spatial distribution are set only once in this step. The following steps are repeated in loop until reaching the WRO state.

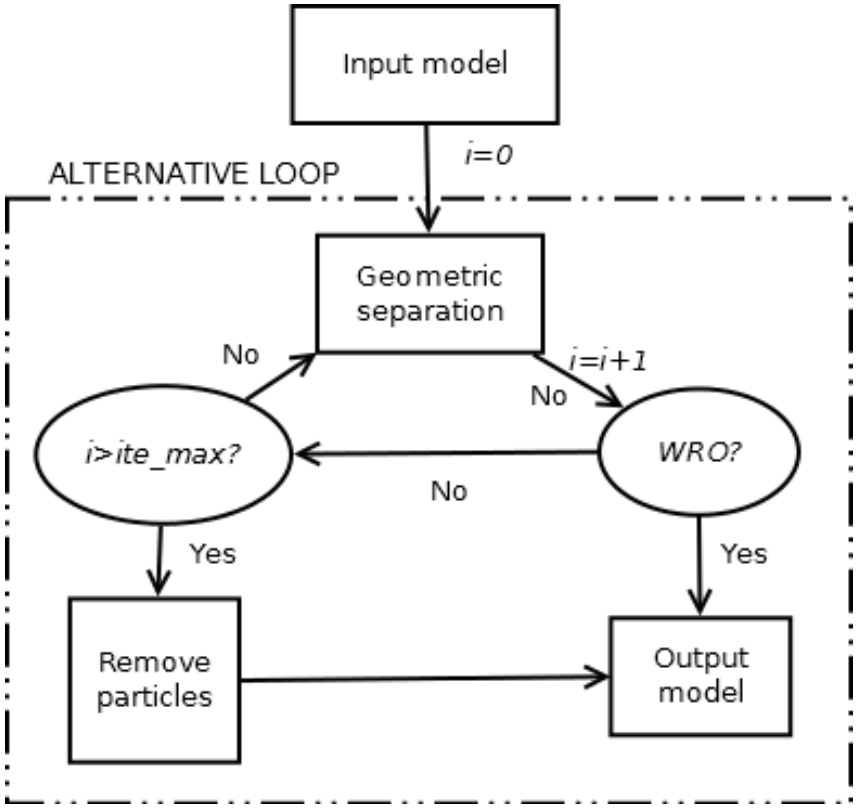
These following steps of the algorithm are strategically used and repeated to reduce these overlaps: geometric separation, particle reallocation, particle removal and particle insertion. The geometric separation moves the overlapping particles. If the WRO state is not achieved after the geometric separation, the particle reallocation or the particle removing are performed. The reallocation moves particles from most filled areas to empty areas, improving the homogeneity of the model. If the overlaps are not reduced to a satisfactory proportion after several combined tried of geometric separation and reallocation, some particles are removed from the domain. In the figure, the index j computes the number of interactive steps for the particle removal. If the counter j exceeds ite_r, it is necessary to remove particles. After a few steps the removed particles are reinserted into the empty areas on domain. The index i indicates a counter for the total number of geometric separation interactions. If after the limit of reinsertion attempts the overlappings are not satisfactorily mitigated and the i counter exceeds ite_i (maximum number of interactions per inserted particle), then the removed particles stay outside the model. This permanent removal is not desired because it affects the output configuration of the filling ratio and the particle size distribution.

4.2.2 Alternative geometric separation methodology

In order to create a parallel version of the strategy, some steps of the original method are modified or eliminated. The modified methodology is shown in Figure 29.

The step of generating the input model is almost stantaneos when compared with the iterative loop. Because of that, only yhe steps in the loop are executed in GPU. The index i indicates a counter for the total number of geometric separation iterations. If the counter i exceeds ite_{max}, the particles still overlapping particles are removed from the model. The

Figure 29 – Parallel geometric separation adaptation.



generation of the input model, as described in Figure 28, is the only step that is based on CPU serial. The changes in the algorithm are in the iterative loop to allow the parallel execution, and the main changes are explained in the following subsection.

4.2.3 Parallel geometric separation loop

The other steps, that corresponds to the iterative loop in Figure 29, can both be performed in serial or totally in parallel on OpenCL supported devices. In these steps, when using the GPU as processing device, there are no communication between the GPU and CPU.

Both the particle reallocation and insertion steps are removed in the alternative loop in order to limit the maximum number of iterations. These removed steps are designed to ensure greater accuracy when working with models with values of filling ratio above 80%. However, these steps parallel execution is of high computational cost and would compromise the effectiveness of the strategy.

After 1000 iterations, about 1% of the particles present overlapping over the tolerance. By applying a margin of error, the maximum number of iterations is defined as 2000. This value is based in previous studies of the serial strategy, showing a good convergence for most of the evaluated cases.The particles are only removed if the WRO state is not achieved until this iteration limit.

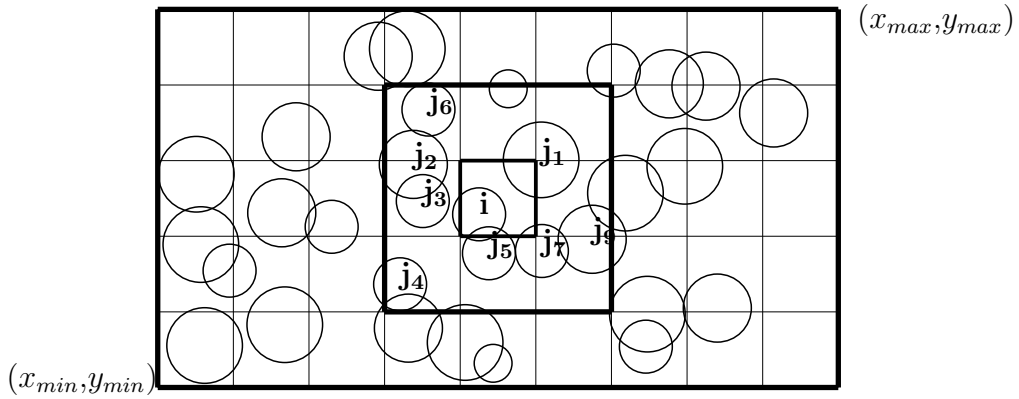
4.2.4 Geometric separation

Geometric separation is the main method for changing particle positions in the proposed strategy. This process aims to eliminate the particle overlaps. Initially it is required to perform domain decomposition into smaller cells to improve search for neighbor particles. This step is implemented using the OpenCL API. OpenCL allows several GPU or CPU devices to be used to execute the same parallel implementation codes without any modification.

4.2.4.1 Mapping particles into spatial cells using parallel binary search

All particles are mapped into spatial cells representing a domain decomposition. This mapping is important for the efficiency of the algorithm and it is a prerequisite to treat overlapping particles. Particles can overlap each other and they can also overlap the domain limits (boundaries). Details are described in Section 4.2.4.2. Checking on particle-by-particle overlap may require a high computational cost as the number of particles increases. Cell mapping algorithms allow you to divide particles into nearby neighborhoods to reduce the search area for overlaps on those neighborhoods rather than the full domain. Figure30 illustrates the space decomposition into cells. The bold smaller square is the cell that contains the particle of interest i, and the larger bold square represents the immediate neighborhood containing the candidate particles j_k that can overlap i. In conclusion, searching for particles and incidences in near neighborhoods is computationally more efficient than searching on full domain.

Figure 30 – Domain divided by cells. The external bold square contains the j particles in the immediate neighborhood for the particle i, and the internal bold square is the cell that contains the particle i.



The literature presents techniques of domain decomposition in identical cubic and square cells, often adopted by Discrete Element Method simulations (MUNJIZA; ANDREWS, 1998; WILLIAMS; PERKINS; COOK, 2004). This work uses the classic algorithm NBS presented by Munjiza et al. (MUNJIZA; ANDREWS, 1998), where the domain area is decomposed on uniform-size square cells. Cell edges measuring two times the larger radii

size of all particles. The NBS decomposition algorithm is conceived and studied to this default cell size. Cells with other sizes may not work properly, presenting nonconformity with the standard setting.

The original serial strategy described in Section 4.2.1 uses a mapping algorithm executed in order to create the initial cell structure that store the identification information (particle ID) for the particles that are inside each cell. Because the algorithm is used in each of many iterations during the packing process, due to the particle position updates, it is necessary to allocate for each cell a vector with enough capacity to store all the particle ID information. In order to avoid reallocation time, the serial algorithm sets a constant vector size to store the particles IDs for all the cells during the entire packing process. The size is settled to a value 20% greater than the larger amount of particles inside a cell in the input model. This is a security value that allows the cells to store more particles than it is necessary. This methodology has the advantage to improve the computing performance, as the cell structure always have vectors of the same size and they do not need to be reallocated. However, this measure is unwise regarding memory usage, as it depends directly on how well distributed are the particles in the input model.

The parallel mapping algorithm uses the NBS algorithm combined with the binary cell searching. Binary cell searching (OOMMEN; PAL, 2014) is used due its good performance in parallel implementations. This method uses two identifier vectors: i) cell identification (cell ID); and ii) related to particle identification (particle ID). Both identifiers relate to each other to allow the algorithm to find all the particles that belong to a given cell.

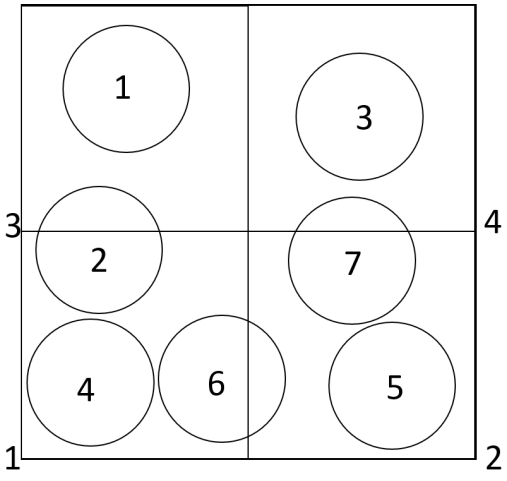


Figure 31 – Particles in a domain divided in four cells

As the binary search is executed only on sorted vectors, the bitonic mergesort method (RANKOVIC; KOS; MILUTINOVIC, 2013) is used prior the searching to sort the ID vectors. This sorting method is well known for presenting good performance in parallel implementation. Both identifier vectors are sorted simultaneously by the *cell id* using a bitonic mergesort variant implemented in this work. The modified algorithm sort both

Table 19 – Binary search vectors

Cell id	**Particle id**
1	2
2	7
1	6
3	1
4	3
2	5
1	4

vectors using only the *cell id* as parameter. Table 20 shows the configuration of the vectors after the dependent sorting.

Table 20 – Binary search vectors after bitoninc dependent sorting.

Cell id	**Particle id**
1	2
1	4
1	6
2	5
2	7
3	1
4	3

The bitonic mergesort is a $O(nlog^2(n))$ comparator algorithm. It is often slower than traditional sorting methods in serial execution that have a $O(nlog(n))$ complexity. However, due tothe good parallel performance, it tends to outperform the serial alternatives depending on the device and the number of elements to be sorted. Table 21 shows the results obtained by kein (2019) with the bitonic mergesort algorithm used to single vectors.

Table 21 – Sorting time in different devices.

Size	**CPU**	**GPU**
4*1024	0.313s	0.434s
64*1024	6.906s	0.872s
256*1024	28.498s	2.018s
2*1024*1024	253.282s	24.68s
64*1024*1024	8861.33s	1192.82s

The CPU used an Intel i7-4790K as CPU and a GeForce GTX 970 as GPU. Table 22 shows the results obtained using the ID dependent bitonic mergesort.

Table 22 – Sorting time in different devices.

Size	**CPU**	**GPU**
4*1024	0.403s	0.397s
64*1024	9.06s	0.811s
256*1024	39.09s	1.821s
2*1024*1024	356.65s	21.79s
64*1024*1024	11417.5s	1040.03s

CPU corresponds to a computer containing a four-core Intel Core i5 processor at a clock speed of 3.2 GHz. GPU is a Nvdia Geforce GTX 1060. This sorting gain depends directly on the number of cores present in the sorting device, as described in the work of Kos (2013). Due to the use of GPUs with different number of cores, the computing time may vary as described in Table 22.

When comparing the binary search information storage efficiency with the original serial method, it is possible to observe a memory gain for variations of the input parameters. This happens because the binary search only depends on the number of particles, while the original searching algorithm also depends on the initial particle distribution on each cell. A simple study varying the filling rate and the number of particles, in a 200x200 unit square box with varied particle sizes. The study using the two searching methods is described in Table 23.

Table 23 – Memory storage usage with the different searching methods.

Filling rate	**Original searching**	**Binary searching**
20%	351 MB	242.5 MB
40%	688.7 MB	475 MB
60%	1.00 GB	722 MB
80%	1.33 GB	970 MB

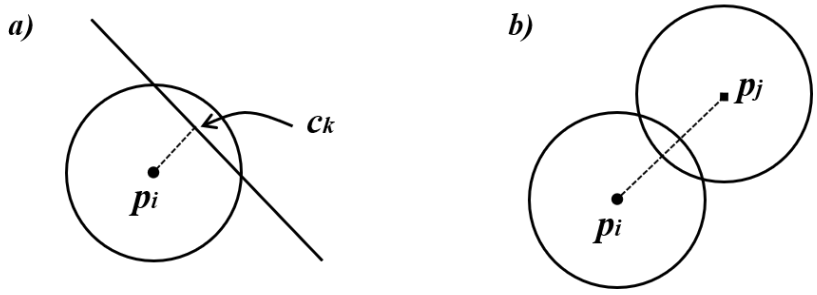
The efficiency use of memory is crucial because GPUs often present a smaller storage capacity than the RAM memory. The original algorithm also may fall in the parallel running condition because it must constantly update the number of particles in each cell. The binary searching is also simpler to be executed in parallel as all the processes are independent to each other.

4.2.4.2 Mapping and eliminating overlapping

An important step for geometric separation procedure is the mapping of overlapping and contacts. Mapping of overlapping particles is performed within each cell. A contact occurs when two particles overlap, ie, the center distances of these particles is lower than the sum of their radii. Contacts can also occur when particles overlap to domain boundary.

These two types of contact interactions are described in Figure 32. Contact mapping is also used in the particles removal and reallocation. Only particles that overlap each other need to be moved and/or removed.

Figure 32 – Representation of the contact interaction. a) contact between a particle and the domain boundary; and b) contact between two particles.



Through an iterative procedure, the geometric separation approach minimizes overlaps by applying incremental displacements. The position of an arbitrary i particle is defined as:

$$\boldsymbol{p_i} = \{x_i, y_i\}, \tag{4.1}$$

x_i and y_i are the Cartesian coordinates. The evolution of these positions along the iterations follows the form:

$$\boldsymbol{p}_i^{n+1} = \boldsymbol{p}_i^n + \boldsymbol{\delta}_i, \tag{4.2}$$

where n indicates the iterative step index and δ_i is the calculated displacement. The calculation of the displacements considers only pairs of particles in contact from a set of n_p elements. The corresponding geometric overlap ϵ_{ij} and displacement increments are calculated for each pair of contacts $[i, j]$. These increments are obtained according to the contact plane normal vector and with the radii of the particles, r_i and r_j, in the form of the following equations:

$$\boldsymbol{\delta p}_i = \sum_{j=1, j\neq i}^{n_p} \Phi_{ij} \frac{\epsilon_{ij}}{||p_i - p_j||}(p_i - p_j), \tag{4.3}$$

$$\boldsymbol{\delta c}_i = \sum_{k=1}^{n_o} \Phi_{ik} \frac{\epsilon_{ik}}{||p_i - c_k||}(p_i - c_k), \tag{4.4}$$

$$\boldsymbol{\delta}_i = \boldsymbol{\delta p}_i + \boldsymbol{\delta c}_i. \tag{4.5}$$

ϵ_{ik} represents the overlapping value between the particles and domain boundaries or obstacles. Its value could be expressed by:

$$\boldsymbol{\epsilon}_{ik} = r_i - ||c_k - p_i||. \tag{4.6}$$

c_k are the coordinates of the contact point, as described in Figure 32. In this work, the only presented obstacle are the domain boundaries, but the method also extends to any

type of obstacle within the domain. The cells that each obstacle belongs to are computed, and the obstacle-particles contact in these cells are mapped. The elements ϕ_{ik} and ϕ_{ij} assume the unit value in case of contact, and the zero value if not. The iterative procedure must occur until the maximum overlap is less than a specified tolerance. The convergence rate is, in general, higher in cases in which the average filling rate is low. The tolerance for the overlap can be defined as a percentage of the particle radii. The particle positions shall not be updated prior to the calculation of the displacements of neighbouring particles. This aids computational parallelization since the calculation of δ_i can be done completely independently.

4.2.5 Removing particles

Particle removal is performed for particles that still present overlapping when the maximum number of iterations is reached. The removal of a particle jeopardizes compliance with the target parameters defined by the input model, because the total number of particles to achieve this goal is generated primarily at input model of particles. Therefore, it is possible to achieve a output model in the WRO state, but that does not meet the desired characteristics targeted. Some methods could be used to improve the achievement of this parameters, such as a response surface intended to mitigate the error. These methods are proposed by the authors in the studies presented in the serial version of the geometric separation methodology.

4.3 Computational performance

This section presents the results obtained using the parallel implementation of the packing strategy. Each subsection discuss a aspect of the result, as well a comparison between the original serial strategy, the alternative strategy executed in serial on CPU, and the alternative strategy executed in parallel on different devices.

4.3.1 Computing time performance

In order to evaluate the computing time of the strategies, two metrics are considered: the total time demanded to generate a granular model and the time per step of the strategy. Due to the fact that the alternative strategy stops at a limit of 2000 iterations as described in section 4.2.3, and the original strategy does not consider a maximum number of steps, it is plausible to say that the alternative strategy tends to execute a smaller number of steps. However, the original strategy presents methods, such as reallocation and reinsertion, that improves convergence. Table 24 presents the results of granular modelling with both strategies, varying the number of particles. All the particles have radii equal to the unit, ant the filling ratio in all the cases is 80%.

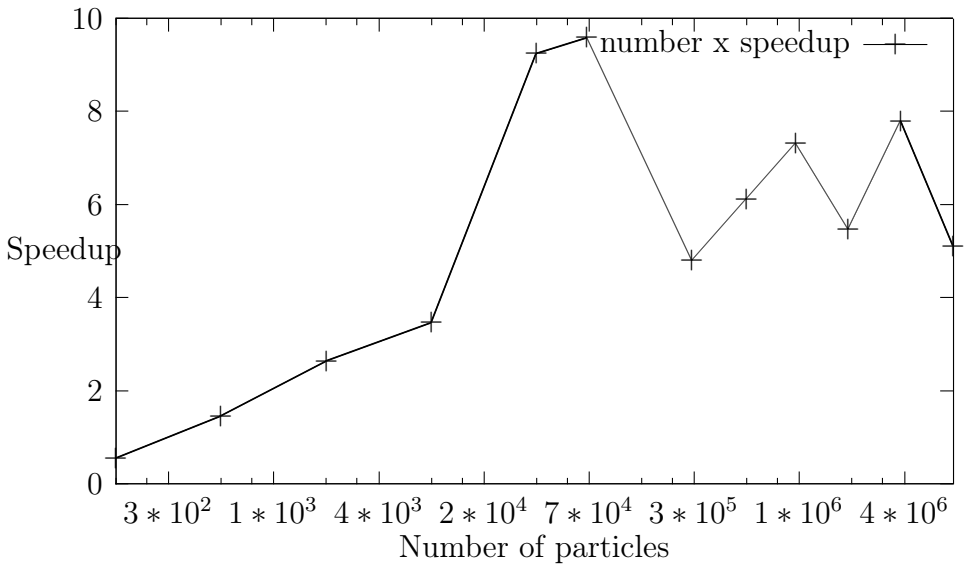
Table 24 – Time profile varying the number of particles with different approaches.

NP	**Original**	**Alternative**	**Parallel CPU**	**GPU I**	**GPU II**
128	0.12s	0.044s	0.93s	0.31s	0.08s
512	0.50s	0.19s	2.99s	0.82s	0.13s
2048	2.12s	0.87s	6.88s	2.13s	0.33s
8192	9.52s	4.44s	17.76s	4.62s	1.28s
32768	50.96s	15.06s	50.58s	11.49s	1.63s
63661	122.3s	32.33s	102.93s	25.99s	3.37s
254647	543.12s	143.73s	209.55s	172.67s	29.97s
524288	1396.28s	412.55s	999.91s	509.01s	67.43s
10^6	3297.14s	1068.06s	2276.65s	747.50s	145.88s
$2 * 10^6$	7128.13s	1602.17s	4692.47s	1543.79s	292.87s
$4 * 10^6$	—	3631.71s	—	2792.12s	465.34s
$6 * 10^6$	—	5992.94	—	4212.50s	1172.45s
$16 * 10^6$	—	—	—	—	2879.01s
$20 * 10^6$	—	—	—	—	4752.01s
$50 * 10^6$	—	—	—	—	14920.6s

The CPU used is an Intel Core i7 four core processor at a clock speed of 3.2 GHz and 16 GB of RAM memory. The GPU I is an Intel Iris with float32 theoretical performance of 806.4 GFLOPS, 40 cores and 1.5 GB of device memory. GPU II is a Nvidia Geforce GTX 1060 with float32 theoretical performance of 4.375 TFLOPS. It has 1280 cores and 6 GB of device memory. The results in Table 24 show that the CPU parallelization does not bring any improvements to the computing time in comparison to the serial execution. A possible reason is the number of CPU cores and the sorting method used by the alternative strategy. The bitonic mergesort has a huge gain of performance when executed in GPU due to the larger number of cores when compared to CPUs. Because the sorting step is executed on every iteration in the alternative geometric separation method, both the original and alternative serial implementations outperform the parallelization on CPUs. The serial strategies outperforms GPU I in the smaller models due to the communication time between the host and computing device. On the other hand, the GPU parallelization brings a huge gain of performance, especially in larger models. As the models increase in size, and because the communication time is constant (existing only because the OpenCL function calls), the results presented by the GPU I shows better time results than both serial implementations. GPU II outperforms all other methods because its the theoretical capacity is several times greater than GPU I, and it has more processing cores. Besides, GPU II is capable to generate larger models than the other used devices. The Figure 33 shows the speedup of the parallel alternative version compared to alternative strategy executed in serial. The behavior of the $SpeedupXN.particles$ shows a peak around 100,000 particles. It is expected such behavior because the bittonic mergesort method employed by the strategy. As described in the work of Kos (RANKOVIC; KOS; MILUTINOVIC,

2013) and by Tab. 22, the speedup of this sort method has a peak and a limit, and they are strictly related to the device where it is executed.

Figure 33 – Speedup of the alternative parallel strategy varying the number of particles.

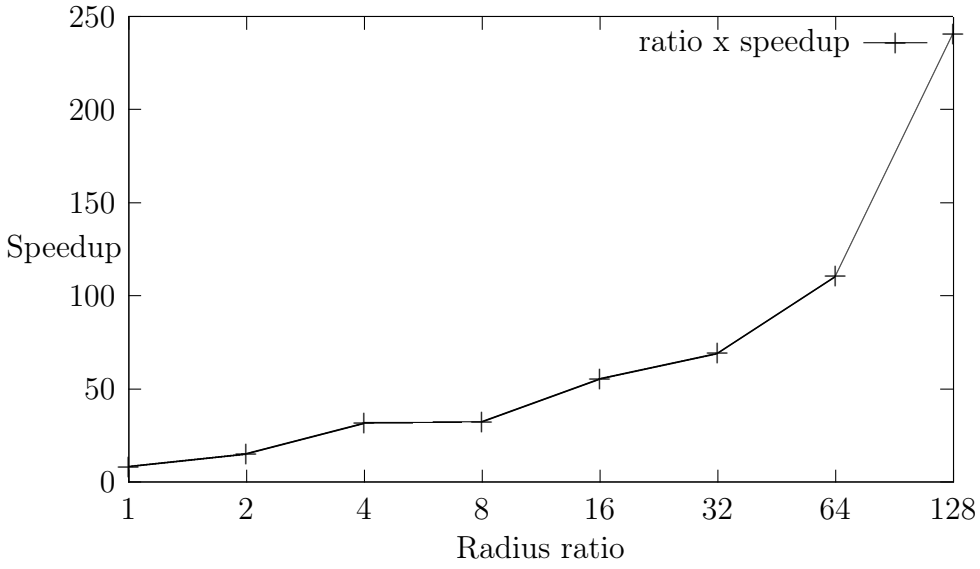


A important factor that influence the computing time of the geometric separation packing strategy is the ratio between the bigger and smaller particle radii. Table 25 shows the computing time of the alternative parallel and serial strategy in the case that there is only one particle larger than the others by a scaling ratio. That is the worst case scenario for the algorithm because the NBS domain decomposition. All the models are composed by 100000 particles, varying only the radius of the bigger particle.

Table 25 – Time profile varying the relation between the the larger and smaller radii.

R_{max}/R_{min}	**Serial**	**Parallel GPU I**	**Parallel GPU II**
1	0.033 (s/ite)	0.023 (s/ite)	0.004 (s/ite)
2	0.09 (s/ite)	0.033 (s/ite)	0.006 (s/ite)
4	0.3175 (s/ite)	0.072 (s/ite)	0.01 (s/ite)
8	1.20 (s/ite)	0.22 (s/ite)	0.037 (s/ite)
16	4.71 (s/ite)	0.934 (s/ite)	0.085 (s/ite)
32	17.96 (s/ite)	2.91 (s/ite)	0.26 (s/ite)
64	64.07 (s/ite)	10.22 (s/ite)	0.58 (s/ite)
128	269.92 (s/ite)	31.93 (s/ite)	1.1206 (s/ite)

Figure 34 – Speedup of the alternative parallel strategy varying the radii ratio.



This results show that the computing time per iteration grows as the scaling rate increases. The main reason for this is how the cell NBS division algorithm works. The value of the maximum radius is directly related to the cell size. Because of that, some cells can contain several smaller particles instead of some big particles. As the radii scaling ratio increases, the algorithm goes from a $O(nlog^2(n))$ complexity to a $O(n^2)$ complexity. GPUs present advantage in more complex algorithms due to the elevated number of cores. Therefore, the speedup of the alternative parallel strategy is often greater than that presented by Figure 33, because of the different radii sizes in regular granular systems.

Figure 34 shows the speedup comparing parallel execution in GPU II to the serial execution in CPU in the worst case scenario. This figure shows that the maximum speedup also depends of the characteristics of the models, and the parallel implementation out-speed the serial in most of the cases.

4.3.2 Algorithm efficiency and validation

Comparing the original to the alternative algorithm, some steps were removed or changed in order to allow the total parallel execution. This means that the original algorithm presents greater precision when reproducing some models. The validation of the alternative method is the reproduction of the examples presented by Frery et al. (FRERY et al., 2011), both with the original and alternative algorithm. Table 26 shows the porosity values of the samples in Frery et al. work obtained when using the different strategies. The porosity (η) value is the completion of the filling rate, as described in the equation:

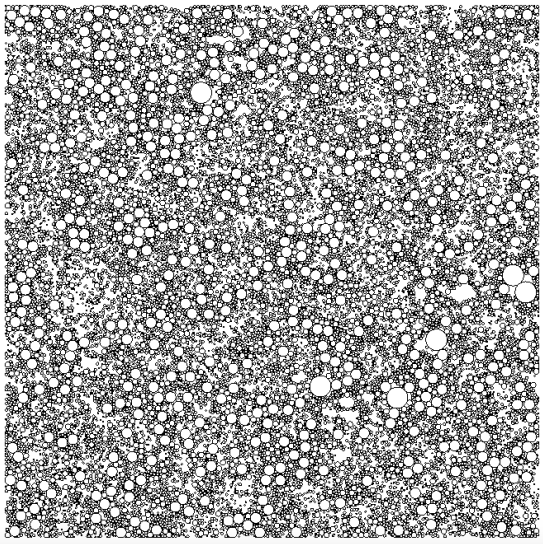
$$F_{rate} = 1 - \eta. \tag{4.7}$$

Table 26 – Porosity values of soil samples and numerical models (FRERY et al., 2011).

Sample	η	η_{Frery}	$\eta_{original}$	$\eta_{alternative}$
1	0.350	0.345	0.353	0.360
2	0.430	0.421	0.430	0.431
3	0.420	0.418	0.431	0.424
4	0.390	0.390	0.395	0.390
5	0.400	0.399	0.403	0.402
6	0.330	0.328	0.339	0.333
7	0.440	0.430	0.439	0.444

Porosity values achieved by the alternative strategy are close to both the experimental and numerical results presented by Frery. The Figure 35 shows the visual representation of the model generated from sample 2 form Frery's work size distribution and porosity data using the parallel geometric separation strategy.

Figure 35 – Model with Frery's sample 2 size distribution and porosity.



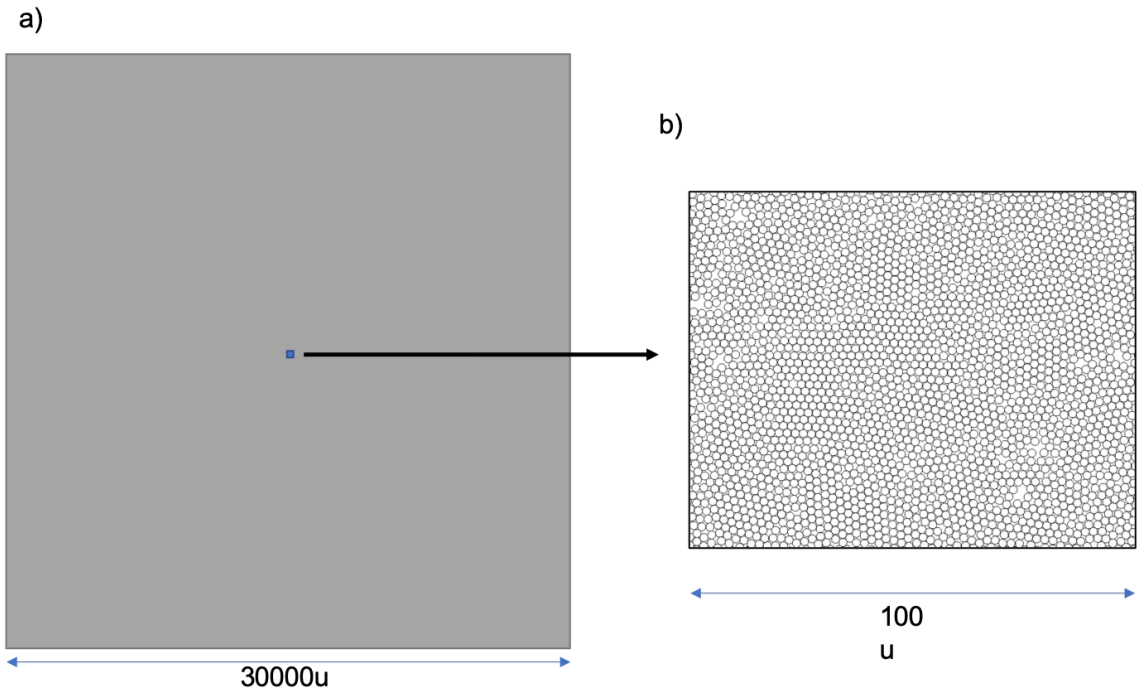
The figure presents a model with no particle overlaps and respecting the parameters specified in Frery's work. In order to illustrate the method capacity, models with elevated particle numbers are generated. The larger model achieved by GPU II contains 50,000,000 particles. A smaller model with 8,000,000 particles are presented in Figure 36, and the data profile about this model is present in Table 27.

Table 27 – Data profile of the model with 8,000,000 particles.

Porosity	**20%**	**20.0007%**
Particle Number	**8,000,000**	**7,999,733**

The input model has 8,000,000 particles, and the output model has 7,999,733 particles. The output porosity achieved is 20.0007%, a close value despite the number of particles

Figure 36 – Model with 8,000,000 particles. a) scale representation of the entire square domain and b) a smaller fraction with details.



removed. The profiles shows that both the size distribution and the target porosity are achieved within the expected tolerance.

A model with 50,000,000 was generated executing the parallel algorithm on the Nvidia GTX 1060 graphic. The computing time to generate this model was about 4 hours, and the memory allocated on GPU during the process was 2.2 GB. The recent results presented by Cintra et al. (2016a) work show a model with 7,800,000 particles. Recarey et al. (2019) present a model with about 10,000,000 particles, and Govender, Wilke and Kok (2016) simulated a model with 60,000,000 particles. The results of the present work show that the proposed methodology can generate large scale models comparable to the state-of-the-art from the literature. It is also possible to generate even bigger models with more powerful GPU hardware.

4.4 Commentaries about the technique

It was presented a parallel alternative to the geometric separation particle packing algorithm. The algorithm is able to generate bi-dimensional granular models with disk shaped particles and prescribed input parameters. The output model of particles is obtained by geometric separation of the overlapping particles that are initially generated in random positions.

The computing time improvement can vary depending on the input parameters, but the results show that speedups above 200 times can be reached using a single commercial

video card, such as Nvidia GTX 1060. Using a device with larger processing power, and more computing cores, allows the algorithm to achieve an even better computational performance. Load tests also show that the parallel implementation has a better memory efficiency. This feature allows to generate bigger models than the original strategy could achieve.

The strategy is also validated by comparing the results with the experimental and numerical data in the literature. A natural evolution of the algorithm involves a three-dimensional version of the strategy. However, studies are still needed to obtain a better understanding of the correlation of the parameters used.

This strategy achieves the main objective of generating particle arrangements with a parallel version of the geometric separation method. The parallelization turns possible the achievement of bigger models in smaller amount of time, without compromising the strategy efficiency and accuracy.

5 FINAL CONSIDERATIONS

This work presented three new packing algorithms with prescribed filling rate and particle size distributions.

Distinct response surfaces were needed and obtained for the minimization and geometric separation methodologies. They are implemented to minimize the error associated with the input filling rate and that obtained in the output model. This allows better results without compromising the computing time. All three strategies are validated by comparing the results with experimental and numerical data available in the literature. Examples are also presented to illustrate the applicability of all the strategies and to evaluate their performance.

The distance minimization strategy presents some limitations to non-convex domains. However, the method can achieve filling rate values above 85%, respecting closely the particle size distribution. There is a three-dimensional version of the method that still needs improvement in its accuracy.

The geometric separation strategy also meet the filling ratio and particle size distribution criteria. Despite the strategy does not being able to ensure lower values of porosity than 18%, it can generate several more particles faster than the distance minimization strategy.

The parallel geometric separation methodology does not always achieve good results when the filling rate of the granular model is above 80%. However, this method overcomes the main limitation of the other two that is the computing time demanded to execute some tasks depending on the input model values. Models exceeding 50,000,000 particles, and computing time with speedup above 200 are achieved using the parallelization in GPU. The parallel strategy is capable of generating bigger models in a smaller amount of time, being close to the state-of-the-art present in the literature.

A natural evolution of the geometric separation algorithms involves three-dimensional versions of them. However, studies are still needed to obtain a better understanding of the correlation of the parameters used for three-dimensional models, especially those that used the response surface to mitigate the error. Studies about the parallel implementation of the particle reallocation, reinsertion, and different cell mapping scheme are still object of study to improve the parallel strategy accuracy.

This work achieves the main objective of presenting three particle packing methods and submitting three research papers to renowned Journals. These results bring both good contributions to the particle packing field and the Post-graduate Program in Civil Engineering (PPGEC).

REFERENCES

BAUGH, J.; KONDURI, R. K. S. Discrete element modelling on a cluster of workstations. *Engineering with Computers*, v. 17, 2001.

BROWN, K. et al. Parallel strategies for crash and impact simulations. *Computer Methods in Applied Mechanics and Engineering*, v. 184, n. 2-4, p. 375–390, 2000. ISSN 0045-7825.

CAMPBELL, C.; CLEARY, P.; HOPKINS, M. Large-scale landslide simulations - global deformation, velocities and basal friction. *Journal of Geophysical Research-Solid Earth*, v. 100, n. B5, p. 8267–8283, May 10 1995. ISSN 0148-0227.

CAMPOS, H. F. et al. Determination of the optimal replacement content of portland cement by stone powder using particle packing methods and analysis of the influence of the excess water on the consistency of pastes. *IBRACON*, v. 347, 2019.

CHANG, K.-J.; TABOADA, A. Discrete element simulation of the Jiufengershan rock-and-soil avalanche triggered by the 1999 Chi-Chi earthquake, Taiwan. *Journal of Geophysical Research-Earth Surface*, v. 114, JUL 22 2009. ISSN 0148-0227.

CINTRA, D. T. et al. A hybrid parallel dem approach with workload balancing based on hsfc. *Engineering Computations*, v. 33, n. 1, 2016.

CINTRA, D. T. et al. A parallel dem approach with memory access optimization using hsfc. *Engineering Computations*, v. 33, p. 2463–2488, 11 2016.

CLAUSEN, O. et al. Discrete-element modelling of hanging wall deformation along the d-1 fault t system, danish north sea, with implications for gas chimney interpretations. *Journal of Structural Geology*, v. 118, 2019.

COETZEE, C. Particle upscaling: Calibration and validation of the discrete element method. *Powder Technology*, v. 334, 2019.

CUI, L.; O'SULLIVAN, C. Analysis of a triangulation based approach for specimen generation for discrete element simulations. *Granular Matter*, v. 5, n. 3, p. 135–145, dez. 2003.

CUNDALL, P. A.; STRACK, O. D. L. Discrete numerical-model for granular assemblies. *Geotechnique*, v. 29, n. 1, p. 47–65, 1979.

DAI, L. et al. Dynamics calibration of particle sandpile packing characteristics via discrete element method. *Powder Technology*, v. 347, 2019.

FENG, Y. T.; HAN, K.; OWEN, D. R. J. Filling domains with disks: an advancing front approach. *International Journal For Numerical Methods In Engineering*, v. 56, n. 5, p. 699–713, fev. 2003.

FRERY, A. et al. Stochastic particle packing with specified granulometry and porosity. *Granular Matter*, Springer Berlin / Heidelberg, p. 1–10, dez. 2011. ISSN 1434-5021. Disponível em: <http://dx.doi.org/10.1007/s10035-011-0300-5>.

GASTER, B. et al. *Heterogeneous Computing with OpenCL*. [S.l.: s.n.], 2013.

GOVENDER, N. et al. Discrete element simulation of mill charge in 3d using the blaze-dem gpu framework. *Minerals Engineering*, v. 79, p. 152 – 168, 2015. ISSN 0892-6875. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0892687515300042>.

GOVENDER, N.; WILKE, D.; KOK, S. Blaze-demgpu: Modular high performance dem framework for the gpu architecture. *SoftwareX*, v. 5, 05 2016.

HAN, K.; FENG, Y. T.; OWEN, D. R. J. Sphere packing with a geometric based compression algorithm. *Powder Technology*, v. 155, n. 1, p. 33–41, jul. 2005.

HE, Y.; BAYLY, A. E.; HASSANPOUR, A. Coupling cfd-dem with dynamic meshing: A new approach for fluid-structure interaction in particle-fluid flows. *Powder Technology*, v. 325, 2018.

KEIN, J. *GPU Sorting Algorithms in OpenCL*. Zenodo, 2019. Disponível em: <https://doi.org/10.5281/zenodo.3490924>.

KRUMBEIN, W. C. The sediments of barataria bay. *Journal of Sedimentary Petrology*, v. 7, 1937.

LABRA, C.; ONATE, E. High-density sphere packing for discrete element method simulations. *Communications In Numerical Methods In Engineering*, v. 25, n. 7, p. 837–849, jul. 2009.

LEVENBERG, K. A method for the solution of certain problems in least squares. *Quart. Applied Math.*, v. 2, p. 164–168, 1944.

LIU, L.; ZHANG, Z.; YU, A. Dynamic simulation of the centripetal packing of mono-sized spheres. *Physica A*, v. 268, n. 3-4, p. 433–453, JUN 15 1999. ISSN 0378-4371.

LOPES, L. G. O.; CINTRA, D. T.; LIRA, W. W. M. A geometric separation method for non-uniform disk packing with prescribed filling ratio and size distribution. *Computational Particle Mechanics*, 2019.

LOPES, L. G. O.; CINTRA, D. T.; LIRA, W. W. M. A particle packing parallel geometric method using gpu. *Engineering Computations*, 2019.

LOPES, L. G. O. et al. A particle packing method for non-uniform sizes and prescribed filling ratio. *Engineering with Computers*, 2019.

LOZANO, E. et al. An efficient algorithm to generate random sphere packs in arbitrary domains. *Computers & Mathematics with Applications*, v. 71, n. 8, p. 1586 – 1601, 2016. ISSN 0898-1221. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0898122116300864>.

MARQUARDT, D. W. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, SIAM, v. 11, n. 2, p. 431–441, 1963.

MECHTCHERINE, V. et al. *Simulation of Fresh Concrete Flow*. 1. ed. [S.l.]: Springer, 2014. v. 15.

MUNJIZA, A.; ANDREWS, K. R. F. Nbs contact detection algorithm for bodies of similar size. *International Journal for Numerical Methods in Engineering*, Department of Engineering, QMW, University of London, London, U.K., v. 43, n. 1, p. 131–149, 1998. ISSN 1097-0207.

ONATE, E.; ROJEK, J. Combination of discrete element and finite element methods for dynamic analysis of geomechanics problems. *Computer Methods in Applied Mechanics and Engineering*, v. 193, n. 27-29, p. 3087–3128, 2004. ISSN 0045-7825.

OOMMEN, A.; PAL, C. Binary search algorithm. *International Journal of Innovative Research in Technology*, v. 1, 2014.

PIZETTE, P. et al. Dem gpu studies of industrial scale particle simulations for granular flow civil engineering applications. *EPJ Web of Conferences*, v. 140, p. 03071, 01 2017.

RANKOVIC, V.; KOS, A.; MILUTINOVIC, V. Bitonic merge sort implementation on the maxeler dataflow supercomputing system. *The IPSI BgD Transactions on Internet Research*, v. 9, 07 2013.

RECAREY, C. et al. Advances in particle packing algorithms for generating the medium in the discrete element method. *Computer Methods in Applied Mechanics and Engineering*, v. 345, 2019.

SHAN, Z. D.; ZHU, F. X. A model for predicting dynamic cutting forces in sand mould milling with orthogonal cutting. *Chinese Journal of Mechanical Engineering*, v. 103, 2018.

SIIRIA, S.; YLIRUUSI, J. Particle packing simulations based on newtonian mechanics. *Powder Technology*, v. 174, n. 3, p. 82–92, MAY 25 2007. ISSN 0032-5910.

UDDEN, J. A. Mechanical composition of clastic sediments. *Geological Society of America Bulletin*, v. 22, 1914.

WALTHER, J.; SBALZARINI, I. Large-scale parallel discrete element simulations of granular flow. *Engineering Computations*, v. 26, p. 688–697, 01 2009.

WILLIAMS, J. R.; PERKINS, E.; COOK, B. A contact algorithm for partitioning arbitrary sized objects. *Engineering Computations: Int J for Computer-Aided Engineering*, Emerald Group Publishing Limited, p. 235–248, 2004. ISSN 0264-4401. Disponível em: <http://dx.doi.org/10.1108/02644400410519767>.

XU, L.; LUO, K.; ZHAO, Y. Numerical prediction of wear in sag mills based on dem simulations. *Powder Technology*, v. 329, 2018.