Application of molecular dynamics simulations for the generation of dense concrete mesoscale geometries

Thomas Titscher[a,*], Jörg F. Unger[a]

[a] *BAM Federal Institute for Materials Research and Testing, Unter den Eichen 87, 12205 Berlin, Germany*

Abstract

The problem of polydisperse sphere packings is applied to concrete mesoscale geometries in finite sized specimens. Realistic sphere diameter distributions are derived from concrete grading curves. An event-driven molecular dynamics simulation using growing particles is introduced. Compared to the widely used random sequential addition algorithm, it reaches denser aggregate packings and saves computation time at high volume fractions.

A minimal distance between particles strongly influences the maximum aggregate content. It is essential to obtain undistorted elements when meshing the geometry for finite element simulations. The algorithm maximizes this value and produces meshable concrete mesostructures with more than 70% aggregate content.

Keywords: polydisperse sphere packing, mesoscale geometry, concrete mesostructure, molecular dynamics

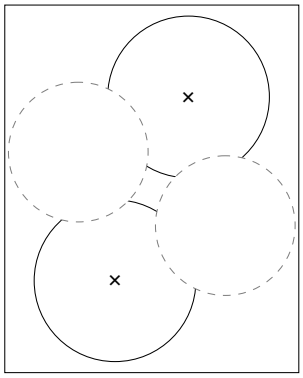
1. Introduction

The random packing of spherical particles is a common problem in research and industry, e.g. packaging, solidification processes and material science. The particle volume fraction ϕ describes the ratio between particle and specimen volume. The maximum value $\phi_{\max}$ for a given configuration depends on the particle size distribution and the specimen size, and is of high interest in these applications. For a random monodisperse size distribution and a infinite specimen, the maximum value $\phi_{\max} \approx 64\%$ is established [1]. Bidisperse and polydisperse size distributions are characterized by numerous parameters. This makes a general understanding of the underlying processes difficult [2] and $\phi_{\max}$ is discussed only for certain types and parameters of size distributions, e.g. log-normal distributions with varying width parameter σ [3].

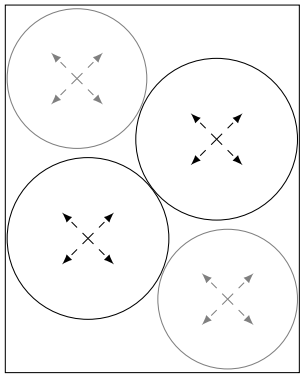
In this paper, the special case of concrete aggregates is investigated. The size distribution is characterized by grading curves that specify the mass fraction of aggregates passing through sieves of varying sizes. An optimization of the grading curve is an important part of modern concrete design [4]. For a thorough understanding of the macroscopic concrete properties, it is important to consider the influence of the heterogeneous structure. For a realistic model of concrete on the mesoscale, the size, position and orientation of aggregates can be either determined experimentally [5] or simulated numerically [6]. Modeling concrete as a three-component composite consisting of aggregates embedded in the mortar matrix with a special modeling of the interface zone makes it possible to use relatively simple constitutive models for each individual component resulting in a complex macroscopic response [7]. In a first approximation, aggregates can be approximated as spherical inclusions.

The focus of the paper is the generation of geometries for particle reinforced composites such as concrete, where the size distribution is polydisperse. Typical volume fractions of concrete particles are in the range of $\phi = 60 - 80\%$, which is relatively close

*Corresponding author. Tel.: +49 30 8104 4551.
Email address: thomas.titscher@bam.de (Thomas Titscher)



(a) The RSA algorithm fixes a particle's positions, once it is placed (solid). All possible positions for new particles (dashed) will overlap with existing ones.



(b) Movable particles allow rearrangements. In combination with growing particles, this can lead to jammed packings.

Fig. 1: Main advantages of moving particle methods over RSA methods at high particle volume fractions ϕ.

to the maximum packing. For the numerical simulation, a two stage *take-and-place* algorithm is commonly used [8, 9]. In the *take* phase, the particles are generated based on the prescribed grading curve, the particle volume fraction ϕ and the specimen volume. In the *place* phase, the particles are positioned within the specimen.

A very common algorithm for the particle placement is the random sequential addition (RSA) algorithm introduced by Widom in 1966 [10]. The particles are randomly placed into the specimen - one after another - starting with the largest ones. Separation checks with previously placed particles are carried out to ensure a non-overlapping packing. There are several ways do deal with overlaps. In the classic algorithm, a particle is only placed into the specimen when it is not overlapping with previously inserted particles - otherwise a new random position is calculated. In a stochastic-heuristic algorithm, the particles are slightly moved [11]. Another alternative is the drop-down algorithm, where a random position at the upper surface of the specimen is picked. The particle falls down and remains where it first touches another particle [12].

The simplicity of random sequential addition and its high efficiency at low particle densities ϕ are huge advantages of this algorithm. The main problem of this algorithm is the fact that a placed particle always remains at its initial position. For high particle densities, it is very likely that previously placed particles block the next ones. The remaining gaps cannot be filled resulting in a loosely packed state (see Fig. 1a).

In 1959, Alder and Wainwright introduced the idea of event-driven molecular dynamics (EDMD)[13] for efficient simulations of molecule interactions in dilute systems. All particles in the system are constantly moving via free-flight dynamics and collisions between them or the walls are fully elastic. The extension of this algorithm to growing particles by Lubachevsky and Stillinger in 1990 allowed to reach a random close packing of discs in 2D [14]. The advantage of this approach is shown in Fig. 1b. Several improvements of the algorithms and growing PC performance allowed the simulation of larger systems in higher spacial dimensions. Kansal et al. used this algorithm to simulated maximally random jammed (MRJ) packings of equal spheres [15]. With varying growth rates, they also investigated bidisperse packings [16]. They used a RSA algorithm for a initial sphere distribution at $\phi = 35\%$ to speed up the simulation. MRJ packings of hyperspheres in $\mathbb{R}^4 \ldots \mathbb{R}^7$ were created by Skoge et al. [1]. Besides for the packing problem, the EDMD algorithm is used in various applications, i.e. shock dynamics [17], homopolymers [18] and hard sphere glasses [19].

There are other approaches for the concrete geometry creation that are mentioned here, but not further investigated. Sonon et al. increased the efficiency of the RSA for two dimensional specimens [20]. Instead of choosing the new particle positions randomly, a level set function is used that indicates valid, non-overlapping positions. A geometric method is used by Jerier et al. [21]. Particles are placed on the nodes and edges of a previously generated mesh. Geometrically complex structures can be filled with this technique. However, the particle size distribution is restricted to the underlying mesh. A popular approach using moving particles is the discrete/distinct element method (DEM). It was introduced by Cundall and Strack and is based on time driven MD [22]. During a fixed time step, overlapping of particles is allowed and results in a separation force that is considered in the next time step. An application to concrete mesostructure can be found in [23].

In this paper, the RSA and the EDMD algorithm for the packing of polydisperse spherical particles is compared. The procedure for the conversion of a mass distribution from real mix designs for concrete into a diameter distribution is presented in Sections 2 and 3. Afterwards, implementation details and a complexity analysis of the RSA and EDMD algorithm are presented in Sections 4 and 5. In a final section, numerical examples are discussed. The EDMD implementation is validated in a monodisperse jammed packing setup. Then, polydisperse size distributions are investigated. The maximal particle volume fraction for both algorithms is determined and the computational efficiency is discussed. The paper concludes with a general comparison of different sampling procedures to build mesoscale models for concrete.

2. Particle size distribution

The aim of the following procedure is to create a random particle *size* distribution that matches a given particle *mass* distribution [7]. This method can easily be applied to given particle volume distributions as well.

The particle mass distribution is given by a grading curve that consists of different mineral size classes i. They contain particle sizes in the diameter range of $d_{\min,i} \leq d < d_{\max,i}$. Their summed mass m_i is calculated with the function $F_m(d)$ in Eq. (1) describing the mass ratio of particles smaller than the diameter d and the mass m_{total} of all particles. Therefore, $F_m(d)$ is a cumulative distribution function (CDF) of the particle mass

$$m_i = m_{\text{total}} \left[F_m(d_{\max,i}) - F_m(d_{\min,i}) \right]. \tag{1}$$

Inside each mineral class i, the mass CDF $\tilde{F}_{m,i}(d)$ is assumed to be linear on a logarithmic scale

$$\tilde{F}_{m,i}(d) = \begin{cases} 0 & \text{if } d < d_{\min,i}, \\ \dfrac{\ln d - \ln d_{\min,i}}{\ln d_{\max,i} - \ln d_{\min,i}} & \text{if } d_{\min,i} \leq d < d_{\max,i}, \\ 1 & \text{if } d \geq d_{\max,i}. \end{cases} \tag{2}$$

This function is monotonically increasing, $\tilde{F}_{m,i}(d_{\min,i}) = 0$ and $\tilde{F}_{m,i}(d_{\max,i}) = 1$. Thus it fulfills the requirements of a CDF inside each mineral size class.

Based on the mass CDF $\tilde{F}_{m,i}(d)$ in the mineral size class i, an equivalent CDF of the particle number $F_{N,i}(d)$ can be computed. Particles are generated according to $F_{N,i}$ until the mass m_i of the mineral size class i is reached.

The derivative of Eq. (2) in the interval $[d_{\min,i}, d_{\max,i}]$ is the probability density function (PDF) of the particle mass $f_{m,i}(d)$

$$f_{m,i}(d) = \frac{\partial \tilde{F}_{m,i}(d)}{\partial d} = \frac{1}{d\left(\ln d_{\max,i} - \ln d_{\min,i}\right)}. \tag{3}$$

Thus, the term $m_i f_{m,i}(d)$ is the expected mass of particles with the diameter d. Dividing this by the mass $m_s(d)$ of a single particle

$$m_s(d) = \varrho \frac{\pi}{6} d^3 \tag{4}$$

leads to the expected number of particles $n(d)$ in the differential interval $[d, d + \mathrm{d}d]$

$$n(d) = \frac{m_i f_{m,i}(d)}{m_S(d)}. \tag{5}$$

In an arbitrary diameter interval $[d_1, d_2]$ with $d_{\min,i} \leq d_1 < d_2 \leq d_{\max,i}$, the total number of particles N is obtained by integrating $n(d)$

$$N(d_1, d_2) = \int_{d_1}^{d_2} n(d)\mathrm{d}d = m_i \int_{d_1}^{d_2} \frac{f_m(d)}{m_s(d)}\mathrm{d}d. \tag{6}$$

The CDF of the particle number $F_{N,i}(d)$ describes the ratio of the number of particles smaller than d to the total number of particles in the mineral size class. Thus, it is defined by

$$F_{N,i}(d) = \frac{N(d_{\min,i}, d)}{N(d_{\min,i}, d_{\max,i})}. \tag{7}$$

Substituting Eqs. (3) and (6) into Eq. (7) yields

$$F_{N,i}(d) = \frac{d_{\max,i}^3 d_{\min,i}^3}{d_{\max,i}^3 - d_{\min,i}^3} \left(\frac{1}{d_{\min,i}^3} - \frac{1}{d^3} \right). \tag{8}$$

The diameter d is a random variable with its CDF $F_{N,i}(d)$. The probability integral transform states that $\mathcal{U} = F_{N,i}(d)$ is uniformly distributed random variable in the interval [0,1]. Consequently, the distribution d can be calculated by the inverse formulation of Eq. (8) as

$$d = F_{N,i}^{-1}(\mathcal{U}) \tag{9}$$

$$d = \frac{d_{\max,i} d_{\min,i}}{\sqrt[3]{d_{\max,i}^3 (1 - \mathcal{U}) + d_{\min,i}^3 \mathcal{U}}}. \tag{10}$$

The total volume within a mineral size class i is obtained by generating uniformly distributed random numbers and calculating the diameter according to Eq. (10) until the total volume of the mineral size class is reached. The corresponding mass distribution is then in accordance with Eq. (2). The last particle of each mineral class exceeds the mass m_i and the mass difference is subtracted from the mass of the next mineral size class. This procedure is summarized in Algorithm 1.

Fuller's curve defines a special CDF for the particle mass

$$P(d) = \left(\frac{d}{d_{\max}} \right)^q, \tag{11}$$

where $d_{\max}$ is the largest particle diameter and $0 < q < 1$ is a parameter that characterizes the shape of the particle distribution. For lower values of q, the mass fraction of smaller particles increases and the distribution gets finer. Special size distributions for concrete on the basis of Fuller's curves with $d_{\max} = 16$ mm are specified in DIN 1045-2: A16 ($q = 0.7$), B16 ($q = 0.35$) and C16 ($q = 0.22$). These discontinuous grading curves are defined piecewise in terms of $d_{\max,i}$, $d_{\min,i}$ and m_i and provide the input for Algorithm 1.

Algorithm 1: Take Phase

Input: grading curve defined by $d_{\max,i}$, $d_{\min,i}$, m_i

Output: array of particle diameters $d[\,]$

foreach mineral size class i **do**

  m_{target} += m_i - m_{diff}

  while $m_{\text{target}} \geq m_{\text{actual}}$ **do**

    $\mathcal{U}$ = uniform random number

    $d[\,] = \frac{d_{\max,i}\, d_{\min,i}}{\sqrt[3]{d^3_{\max,i}(1-\mathcal{U}) + d^3_{\min,i}\mathcal{U}}}$

    m_{actual} += $\varrho \frac{\pi}{6} d^3$

  end

  m_{diff} = m_{actual} - m_{target}

end

In simulations that use Fuller's curve $P(d)$, the algorithm's input is obtained by dividing the continuous curve into sufficiently small classes according to Eq. (1) with $F_m(d) = P(d)$.

However, for further analytic considerations, it is useful to perform the previously shown derivations with the continuous function $P(d)$ in the whole diameter interval $[d_{\min}, d_{\max}]$, instead of splitting it up into small classes. For example, the expected number of particles in this interval is obtained by evaluating Eq. (6)

$$N(d_{\min}, d_{\max}) = m_{\text{total}} \int_{d_{\min}}^{d_{\max}} \frac{1}{m_s(d)} \frac{\partial P(d)}{\partial d} \mathrm{d}d \tag{12}$$

$$= \phi V \frac{6q}{\pi d^q_{\max}(q-3)} \left(d^{q-3}_{\min} - d^{q-3}_{\max} \right). \tag{13}$$

All the particle size distributions used in this paper are shown in Fig. 2.

3. Manipulation of the mass CDF

The simulation of the total particle mass m_{total} is impossible, since the number of particles tends to infinity for $d_{\min} \to 0$. Consequently, the particle mass CDF must be cut off at $d_{\min} > 0$ and the mass m_{cutoff} of all particles smaller than $d_{\min}$ is neglected.

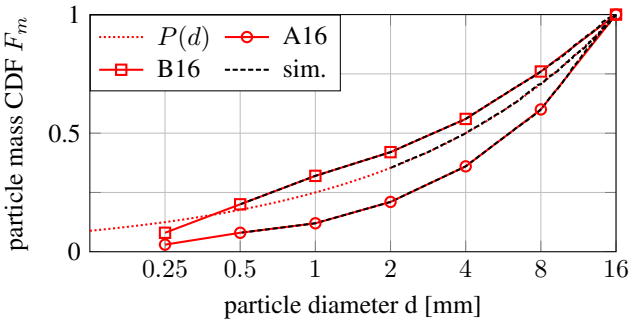


Fig. 2: The simulated particle size distributions (dashed black lines) obtained by Algorithm 1 are in agreement with the red target curves. A16 and B16 are specified in DIN 1045-2, the markers show the limits of the mineral size classes. $P(d)$ is Fuller's curve (Eq. (11)) with $q = 0.5$, $d_{\max} = 16$ mm.

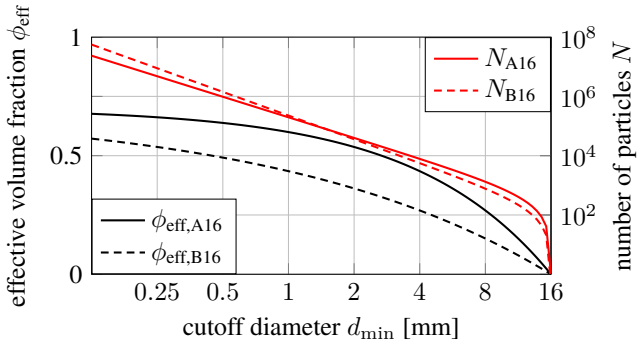


Fig. 3: The piecewise defined grading curves A16 and B16 are approximated with Fuller's curves with $q = 0.7$ and $q = 0.35$. The effective particle volume fraction ϕ_{eff} defined in Eq. (17) decreases with growing $d_{\min}$ since all the particles smaller than $d_{\min}$ are neglected. The number of particles $N = N(d_{\min}, d_{\max})$ is determined analytically by Eq. (13) for a cubic specimen with an edge length $a = 150$ mm and the mixture particle volume fraction $\phi_M = 70\%$.

According to Eq. (1), this is obtained from

$$m_{\text{cutoff}} = m_{\text{total}} F_m(d_{\min}). \tag{14}$$

Thus, a distinction between the particle volume fraction ϕ_M according to the mix design

$$\phi_M = \frac{m_{\text{total}}}{\varrho V} \tag{15}$$

and the effective volume fraction ϕ_{eff}

$$\phi_{\text{eff}} = \frac{m_{\text{total}} - m_{\text{cutoff}}}{\varrho V} \tag{16}$$

$$= \phi_M \left[1 - F_m(d_{\min})\right] \tag{17}$$

is needed, where ϱ is the particle density and V the specimen volume.

Depending on the area of application, either ϕ_M or ϕ_{eff} are of interest. For the simulation of concrete on the mesoscale, the total volume fraction ϕ_M of the mixture is important, since this value is the prescribed parameter in the mix design. However, packing algorithms discussed in literature usually use the effective volume fraction ϕ_{eff}. In the following sections, it is important to carefully distinguish between these values.

In concrete geometry simulations, a common choice for the cutoff diameter is $d_{\min} = 2$ mm [7]. All particles smaller than 2 mm are sand grains that are assumed to be represented by the homogenized mortar matrix. According to Eq. (17), their mass is subtracted from the total mass and the effective volume fraction ϕ_{eff} reduces. In the common case of $\phi_M = 70\%$, which represents the aggregate volume fraction that is physically filled in the concrete mixer, and the grading curve B16, the volume fraction that is actually resolved by the mesoscale geometry model is $\phi_{\text{eff}} = 40.6\%$. For a cubic specimen with an edge length of $a = 150$ mm, the number of simulated particles reduces from infinity ($d_{min} = 0$) to $N \approx 40\,000$ ($d_{min} = 2\text{mm}$). For the grading curve A16, the corresponding values are $\phi_{\text{eff}} = 53.3\%$ and $N \approx 44\,000$. In this case, the geometry creation of A16 is more challenging than the one for B16, since the relative particle volume is larger and more particles have to be simulated. A more general dependency of the cutoff diameter $d_{\min}$ on ϕ_{eff} and N is shown in Fig. 3 for the corresponding Fuller's curve approximations of A16 and B16.

Another problem is related to the meshing of the generated particle geometries and the subsequent simulation using finite elements. For numerical reasons, highly distorted elements should be avoided. Especially for dynamic simulations of concrete using

explicit time integration schemes, like wave propagation or dynamic crack propagation, the critical time step is strongly influenced by the smallest element length. For this purpose, a minimal distance Δd between any two particles is enforced. Consequently, an increase in Δd allows a coarser spatial and temporal discretization of the governing equations. Physically, this minimum distance can be interpreted as a thin mortar film around each particle [24]. Experiments show a coating of mortar around each particle. Schlagen and van Mier suggests a minimal distance $\Delta d = 0.1(d_1 + d_2)/2$ between the particles with diameters d_1 and d_2 [25]. Wang et al. uses the minimal distance $\Delta d = \gamma \min(d_1, d_2)$ and γ depends on the particle volume fraction [8]. In this study, $\Delta d = d_{\min}/2$ is used, unless stated otherwise.

4. Random sequential addition

The RSA algorithm is commonly used for the geometry creation of the concrete mesostructure. Our implementation of the algorithm is as follows. All particles are sorted according to their volume and the place process is started with the largest particle. Uniform random numbers are generated for all coordinates. In the case of axis-aligned box-shaped specimens, boundary collisions can be eliminated by the right choice of the random number interval. For other specimen shapes such as cylinders, a check for boundary collisions is performed.

The check for collisions with already placed particles is done by a simple sphere separation check. If the particle overlaps with previously placed particles, a new random position is chosen. After N^F failed placement attempts of a single particle, the algorithm aborts (stopping criterion).

The complexity of the collision checks of a single particle is $\mathcal{O}(N)$, since it must be checked against all N already placed particles. With a cell method, these numerical costs can be reduced. The specimen is divided into $N_{\text{cell}} \approx N$ cells and the collision checks are only performed within a small number of cells. In the best case of monodisperse particles, each cell contains about one particle and a constant complexity of $\mathcal{O}(1)$ can be reached.

The high polydispersity of concrete particles would cause performance problems, if the cell size is kept constant. A coarse cell structure causes an increased number of particles per cell and lowers the efficiency. A fine cell structure can ensure one particle per cell. However, the addition of a particle that is much bigger than a single cell requires a collision check in many cells.

In the current implementation, this problem is addressed by using a variable cell size. The place process starts with the largest particles and a large cell size related to the particle diameter. Consequently, about one particle fits in a cell and a near optimal performance is achieved. As the particles get smaller, the number of particles per cell increases and the method would become inefficient. Thus, when the particle diameter reaches half of the starting particle diameter, all the cells are rebuild with a smaller cell size. The previously placed particles are added to the new cells and the place process continues. Despite the overhead for adding existing particles in the newly built cells, this technique adapts the cell size to the particle size and ensures only a small number of particles per cell.

5. Event-driven molecular dynamics

Lubachevsky and Stillinger used a two dimensional EDMD algorithm with growing particles to simulate dense monodisperse circle packings. Kansal et al. used a modified version with the following changes [16]. First, they transferred the problem to spheres in the three dimensional space. Second, instead of starting with zero volume spheres, they used an RSA algorithm for an initial

spacial distribution at $\phi_M \approx 35\%$. Third, the growth rate for each sphere was set to be proportional to its initial diameter. With these modifications, bidisperse and polydisperse sphere distributions were simulated in periodic specimens.

In our implementation, rigid walls are used as boundaries. A final size distribution is calculated for a given set of parameters defined by the grading curve, the specimen size and the particle volume fraction ϕ_M. As stated above, the RSA excels at lower particle volume fractions. Therefore, the final particle diameters are reduced, hence the volume fraction is reduced as well. The modified particles are efficiently placed into the specimen using the RSA algorithm to provide an initial spacial particle distribution.

A random initial velocity and a growth rate is given to each particle and the system evolves in time. The growth rate of each individual particle is set in a way that all particles reach their final diameter at the same time t_{end}. Consequently, the resulting size distribution is in accordance with the desired grading curve.

The details of the diameter reduction and the choice of the growth rate vary in the numerical experiments and are discussed in Section 6. In the following, the implementation of our EDMD algorithm is shown in detail.

5.1. Collidable description

All the physical objects of the simulation are organized in *collidables* C. They are either static planar walls W or movable spherical particles P. Cylindrical walls are also implemented. Their collision detection and collision physics are a combination of P and W.

The description of the walls surface W is in point-normal form - $\mathbf{w}_0$ is one point of the wall and $\mathbf{n}_W$ a normalized vector perpendicular to the wall, pointing inwards. Every point $\mathbf{w}$ on the wall satisfies

$$(\mathbf{w} - \mathbf{w}_0) \cdot \mathbf{n}_W = 0. \tag{18}$$

The time dependent particle position $\mathbf{p}(t)$ of a particle P depends on its current position $\mathbf{p}_0$ and its velocity $\mathbf{v}$. The particle is constantly growing, and its radius $r(t)$ depends on an initial radius r_0 and its growth rate γ

$$\mathbf{p}(t) = \mathbf{p}_0 + \mathbf{v}\, t \tag{19}$$

$$r(t) = r_0 + \gamma\, t. \tag{20}$$

5.2. Collision detection

The time until impact for two collidables C^a and C^b is calculated by solving for the distance function $\mathcal{D}$ to vanish

$$t_c \leftarrow \mathcal{D}(t_c, C^a, C^b) = 0. \tag{21}$$

Particles are the only movable collidables. Thus, the first collidable C^a of every collision is a particle ($C^a \rightarrow P^a$). For the particle-particle collisions ($C^b \rightarrow P^b$) the distance function $\mathcal{D}$ is

$$\mathcal{D}(t, P^a, P^b) = \left|\mathbf{p}^a(t) - \mathbf{p}^b(t)\right| - \left(r^a(t) + r^b(t)\right). \tag{22}$$

It results in a quadratic equation for t_c. The distance function $\mathcal{D}$ for the particle-wall collisions ($C^b \rightarrow W^b$) reduces to a linear equation

$$\mathcal{D}(t, P^a, W^b) = \left(\mathbf{p}^a(t) - \mathbf{w}_0^b\right) \cdot \mathbf{n}_W^b - r^a(t). \tag{23}$$

5.3. Collision physics

The collision of two collidables causes their velocities to change. During wall collisions, the incoming velocity vector $\mathbf{v}$ is reflected at the walls normal vector $\mathbf{n}_W$ and results in the outgoing vector $\mathbf{v}'$

$$\mathbf{v}' = \mathbf{v} - 2\,(\mathbf{v} \cdot \mathbf{n}_W)\,\mathbf{n}_W. \tag{24}$$

The three dimensional particle-particle collisions are reduced to one dimensional collisions of their normal velocities v_n. Their tangential velocities $\mathbf{v}_t$ remain unchanged

$$v_n = (\mathbf{n} \cdot \mathbf{v}) \quad \text{with} \quad \mathbf{n} = \frac{\mathbf{p}_0^a - \mathbf{p}_0^b}{|\mathbf{p}_0^a - \mathbf{p}_0^b|} \tag{25}$$

$$\mathbf{v}_t = \mathbf{v} - v_n \mathbf{n}. \tag{26}$$

The post-collision normal velocities v_n' are obtained by a fully elastic collision depending on the particle masses m according to Eq. (4)

$$v_n'^{a,b} = 2\frac{m^a v_n^a + m^b v_n^b}{m^a + m^b} - v_n^{a,b}. \tag{27}$$

The new velocities $\mathbf{v}'$ are then given as

$$\mathbf{v}' = \mathbf{v}_t + v_n' \mathbf{n}. \tag{28}$$

An alternative for calculating the post-collision normal velocities v_n^* is used by Kansal et al., where the sphere masses are assumed to be equal and Eq. (27) reduces to

$$v_n^{*a,b} = v_n^{b,a}. \tag{29}$$

However, they suggest that this detail does not influence the generated packings [16].

5.4. Algorithm

The main loop of an EDMD program is shown in Algorithm 2. The previously defined operations for the collision detection and the collision processing are nested in several loops. Additional operations are required for the handling of an event list and the system update after each collision.

The triple $\langle t_c, C^a, C^b \rangle$ is called event and describes a single collision. All collisions are handled in an event list that is ordered by the collision time t_c. The components of this list are addressed by $\langle t_c, C^a, C^b \rangle_i$. Even though at least one of the collidables C^a or C^b is a particle P, the more general notation is used here.

If the simulation reaches the time $t = t_{\text{end}}$, the desired particle size distribution will be reached. If the change of the simulation time Δt is smaller than a value of ε during 1000 events, the simulation will abort unsuccessfully.

The single steps of Algorithm 2 (basic implementation) including the optimization are implemented as follows:

Input: Before the main loop starts, an initial event list is built by cross checking every particle against every other particle. A binary tree is used for the event list.

In the basic implementation, the number of events per particle depends on the number of particles N. Thus, the length of the event list is in the order of $\mathcal{O}(N^2)$. The use of the **cell method** (explained in Step 5) reduces the number of events per particle to a constant value and the length of the event list to $\mathcal{O}(N)$.

Algorithm 2: Basic implementation

Input: Event list as $\langle t_c, C^a, C^b \rangle_i$

while $t_c < t_{\text{end}}$ **do**

Step 1:Find lowest time t_c and the corresponding event $\langle t_c, C_c^a, C_c^b \rangle$

Step 2:Update system to time t_c

foreach Particle P **do**
$\mathbf{p}_0 = \mathbf{p}_0 + \mathbf{v}_0 t_c$

end

Step 3:Perform collision $\langle t_c, C_c^a, C_c^b \rangle$

Step 4:Delete invalid events of C_c^a and C_c^b

foreach $\langle t, C^a, C^b \rangle_i$ **do**

if C_c^a or C_c^b is in $\langle t, C^a, C^b \rangle$ **then**

delete $\langle t, C^a, C^b \rangle$

end

end

Step 5:Predict new collision times t_c of C_c^a and C_c^b

foreach collidable C^i **do**

solve distance equation:

$t_c^a \leftarrow \mathcal{D}(t_c, C_c^a, C^i) = 0$

$t_c^b \leftarrow \mathcal{D}(t_c, C_c^b, C^i) = 0$

Step 6: Store the events $\langle t_c^a, C_c^a, C^i \rangle$ (sorted)

Store the events $\langle t_c^b, C_c^b, C^i \rangle$ (sorted)

end

end

Step 1 Find next event: Since the event list is sorted by time, the first element in this list is the next event.

Step 2 System update: The position of all spheres at the current simulation time must be known for the collision checks. In a synchronized system, like in the basic implementation, all particle positions are updated after each collision. This causes an additional loop over all particles with memory writes. In a **delayed state** simulation, the particles are usually out of sync [26]. They store the time of their last update and at each collision check their position is recalculated. The cost intensive memory writes for every particle are omitted.

Step 3 Collision: The collision is performed according to the collision physics in section 5.3. In the **delayed state** simulation, the two colliding particles must be synchronized beforehand.

Step 4 Event deletion: All stored events that contain the previously collided particles are now invalid and must be deleted. It is required to search the entire event list for these events. Since the event list is sorted by time and not by particles, the tree structure of the event list cannot be used.

In our implementation, all particles list their events in a **local event list** (with constant length). These local events are then deleted from the global event list. Now, the time-based tree structure of the global event list can be used, since the collision time is known from the local event. Thus, the deletion takes $\mathcal{O}(\log N)$ time per event.

Step 5 Event prediction: New collision times for the collided particles must be calculated. In the basic implementation, this requires a collision check with all other particles and a complexity of $\mathcal{O}(N)$.

In the paper of Alder and Wainwright in which the EDMD was introduced, they also proposed the cell method for more efficient collision checks [13]. The specimen is divided into much smaller disjunct cells that keep track of the particles inside. If a sphere is in the transition from one cell to another, it is temporarily owned by two (or more) cells. Collision checks are only performed inside a cell. Since the number of cells is chosen to be in the order of the particle number, the numerical costs for collisions checks are constant ($\mathcal{O}(1)$). The overhead caused by cell transfer events reduces as particles grow. In a densely packed state, the marginal sphere movement causes almost no transfer events.

Step 6 Event storage: The insert operation has a $\mathcal{O}(\log N)$ complexity in the tree-like event list. The overall complexity depends on the number of new events ($\mathcal{O}(N \log N)$ or $\mathcal{O}(\log N)$ with the **cell method**).

The complexity of each step is summarized in Table 1. With all these optimization, the algorithm has a theoretical overall complexity of $\mathcal{O}(\log N)$. This is in agreement with numerical simulations. A system of N monodisperse particles with the same growth rate developed for five million events to ensure a quite dense packing. Afterwards, the time for the next one million events was measured, averaged and the results are shown in Fig. 4.

Inserting and deleting events consumes a lot of time, depending on the length of the event list. Thus, a **time barrier** is used to keep the event list short. Events that occur after a certain time barrier are not inserted into the event list. When the simulation time reaches the time barrier, the event list is cleared and rebuild. With the right choice of the time barrier, the overhead of rebuilding the entire event list can be compensated and the overall performance increases significantly (by a constant factor). In our simulation, the time barrier was chosen to be approximately 10 times the time required to rebuild the event list.

5.5. Further optimization

For experiments with a system size of about ($\approx 10^4$ particles), the performance of the presented algorithm $\mathcal{O}(\log n)$ is satisfying. More advanced optimization techniques to boost the algorithms performance for larger systems are

Table 1: Complexity of the process of a single event based on the number of particles N. The basic implementation of Algorithm 2 is compared to the optimized one used in this paper.

		complexity	
Step	Task	basic	optimized
1	Find next event	$\mathcal{O}(1)$	
2	System update	$\mathcal{O}(N)$	0
3	Collision	$\mathcal{O}(1)$	
4	Event deletion	$\mathcal{O}(N^3)$	$\mathcal{O}(\log N)$
5	Event prediction	$\mathcal{O}(N)$	$\mathcal{O}(1)$
6	Event storage	$\mathcal{O}(N \log N)$	$\mathcal{O}(\log N)$

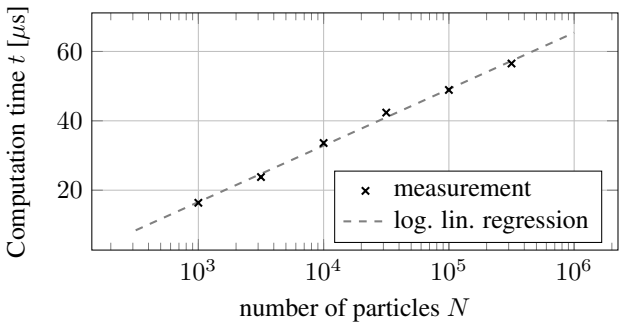


Fig. 4: A system of N monodisperse particles with the same growth rate develops for five million events to ensure a quite dense packing. The time t for the next one million events is measured, averaged and shown in this diagram. Our algorithm shows $\mathcal{O}(\log N)$ scaling for the process of a single event.

near neighbor list: The cell method is most efficient with exactly one sphere per cell. In the case of high polydispersity, this cannot be ensured. Therefore, more efficient techniques, e.g. the near neighbor list [27], are available.

event list: The event list handling (Step 4+6) is the only task with non-constant complexity. With the use of a calendar queue for the event handling, all operations on this queue require a constant time $\mathcal{O}(1)$ [28]. This allows for the implementation of EDMD algorithms, e.g. DynamO[29], where the execution of a single event is independent of the system size.

6. Numerical experiments

In this section, a validation of the EDMD is shown using the example of monodisperse sphere jamming. Next, a comparison between the RSA and EDMD algorithm is performed in two different numerical experiments. In the first one, the maximal particle volume fraction ϕ_{eff} for a given Fuller's curve is investigated. A real concrete specimen with grading curve A16 and a defined distance Δd between the particles is studied in the second experiment. Afterwards, the maximum value of Δd is investigated for A16 and B16 in combination with realistic volume fractions.

All calculations are performed on a single Intel® Xeon® E5-2630L CPU at 2.00 GHz.

6.1. Monodisperse particle jamming

The densest packing of equal (monodisperse) spheres can be achieved in regular lattices. The hexagonal close-packed (hcp) and the face-centered cubic (fcc) arrangements both reach $\phi \approx 74\%$. A maximally random jammed (MRJ) packing describes the closest packing of randomly arranged equal spheres. Unlike the long-range order of the lattice structures, the MRJ consists of short-range ordered clusters with changing orientations. For a three dimensional cell with periodic boundaries, the value of $\phi_{\text{MRJ}} \approx 64\%$ has been verified numerically [15, 1].

In our simulations, rigid boundaries for a realistic representation of the wall effect in real concrete specimen are used. As a consequence, the particle density at the wall is $\phi_{\text{Wall}} = 0$, which reduces the maximum particle density. The influence of this effect decreases with a higher number of particles.

In the validation experiment, the initial particle distribution for the EDMD is obtained using the RSA algorithm with $\phi = 30\%$. The spheres grow with the same growth rate until the simulation aborts and a final particle volume fraction ϕ_{eff} is reached. The results of these simulations for varying particle numbers N and a constant sized cubic specimen are shown in Fig. 5.

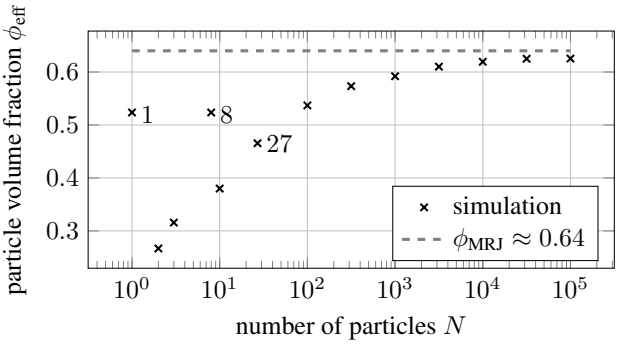


Fig. 5: Jamming of N monodisperse particles. The simulations approach the MRJ limit with increasing number of particles.

A single particle can grow until its diameter reaches the edge length of the cubic specimen and the theoretical volume fraction $\phi = 4/3\pi(0.5)^3 = \pi/6 \approx 52\%$ is reached. For two particles, the densest arrangement is the placement in opposite corners of the specimen. Since the other corners are empty, this causes a drop in the volume fraction. The same effect occurs for three or more particles, but its influence reduces with higher number of particles. At $N = 2^3$, the particle arrange in a simple cubic lattice that has the same ϕ as the single particle. This ordered state is lost for $N = 3^3$. In general, the particle volume fraction increases with the number of particles and gets close to ϕ_{MRJ}. Additionally, these experiments show that the algorithm's performance is good enough to reach a very jammed state.

6.2. Polydisperse particle jamming

The aim of the following numerical experiment is to compare the maximally reachable particle volume fraction and the performance of the RSA and the EDMD algorithm. The polydisperse particle distribution is defined by Fuller's curve $P(d)$ with $q = 0.5$ and $d_{\max} = 16$ mm. A minimal distance of $d_{\min} = 2$ mm is chosen to ensure a reasonable number of particles for the simulation. Since the maximal particle density is investigated, the effective volume fraction ϕ_{eff} is considered.

With increasing particle volume fraction ϕ_{eff}, both algorithms are more likely to fail. This can be expressed as a success rate $\psi(\phi_{\text{eff}})$. A successful attempt is defined by not reaching a stopping criterion. Beginning at $\phi_{\text{eff}} = 55\%$, the particle volume fraction is increased in steps of $\Delta\phi_{\text{eff}} = 0.5\%$. 50 random particle size distributions are calculated for each ϕ_{eff} and are used for both the RSA and the EDMD algorithm. The total computation time of the 50 simulations is divided by the number of successful algorithm runs to get the average time for a successful attempt t_ψ.

The RSA algorithm is applied straightforward with the stopping criteria, $N^F = 10^5$ or $N^F = 10^6$ (see Section 4). In the EDMD algorithm, the final particle diameters d from the particle size distribution are scaled down by a factor of $\delta = 0.90$ to the initial diameters $d_0 = \delta\, d$. The scaled particles are now placed into the specimen using the RSA method. A relative growth rate $\gamma_r = \gamma\, d_0$ with $\gamma = 0.1\,\text{s}^{-1}$ is applied to each particle. If the simulation reaches the time t_{end}, the grown diameters will match the desired particle distribution and the run is successful

$$t_{\text{end}} = \frac{1-\delta}{\delta\gamma}. \tag{30}$$

If the change in the simulation time Δt is below 10^{-6} during 1000 events or the computation wall time reaches 20 minutes, the algorithm will fail.

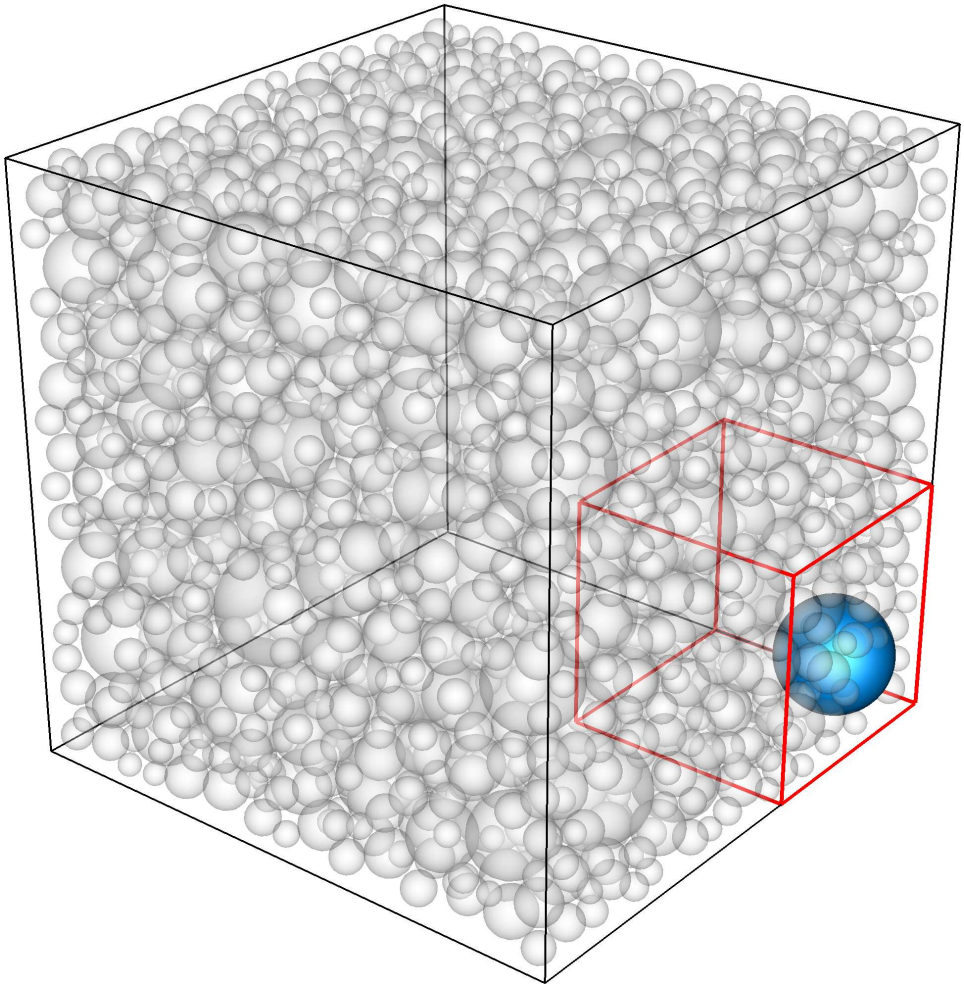


Fig. 6: The particle with the biggest diameter $d_{\max}$ is compared to the two specimen sizes used. The small box has an edge length $a = 2\ d_{\max}$, for the large box $a = 5\ d_{\max}$.

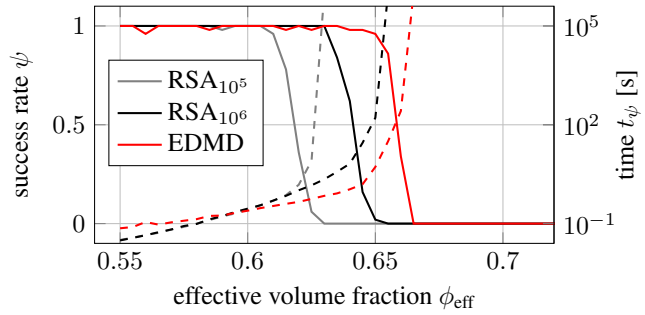
All the simulations are carried out for two different cubic specimens. The edge lengths a are chosen to be multiples of the largest diameter $d_{\max}$. The two specimens with $a = 2d_{\max} = 32$ mm and $a = 5d_{\max} = 80$ mm are shown in Fig. 6. The initial velocities for the EDMD simulations were uniformly distributed between $-0.5 \ldots 0.5\,\mathrm{mm/s}$ in each coordinate.

The results of the simulations are shown in Fig. 7. The success rate is close to $\psi = 1$ for low particle volume fractions. At a certain volume fraction, the success rates drop to $\psi = 0$ and the average time for a successful attempt t_ψ increases. For both specimen sizes the EDMD method reaches the highest particle volume fraction and the RSA method with $N^F = 10^5$ the lowest one. After a break-even point $(\phi_{\text{eff}}, t_\psi) \gtrsim (60\%, 0.2\,\mathrm{s})$ in Fig. 7a and $(\phi_{\text{eff}}, t_\psi) \gtrsim (62.5\%, 4\,\mathrm{s})$ in Fig. 7b, the EDMD algorithms performs faster than the RSA algorithm.

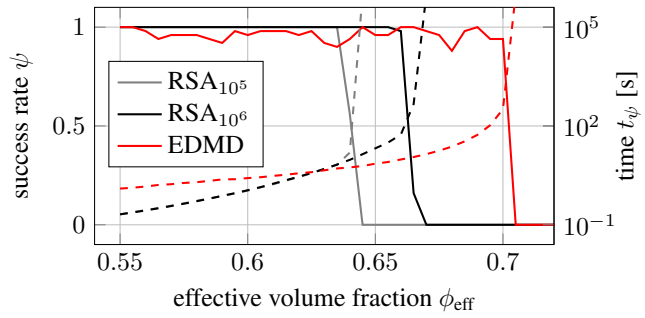
For the larger specimen, the maximal volume fraction $\phi_{\text{eff,EDMD}} = 70\%$ is 5.3% higher than $\phi_{\text{eff,RSA}} = 66.5\%$.

The overall reachable particle volume fraction for a specimen size of 32 mm illustrated in Fig. 7a is below the one for a specimen size of 80 mm shown in Fig. 7b for both algorithms. This is due to the wall effect. The ratio between the specimen volume and the wall area is proportional to the edge length a. Therefore, the influence of the wall effect decreases with larger specimens.

Even at low values of ϕ, the EDMD algorithm fails at some runs. The failure is caused by the stopping criterion $\Delta t < \varepsilon$ in an



(a) Specimen size $a = 32$ mm.



(b) Specimen size $a = 80$ mm.

Fig. 7: Results of the polydisperse particle jamming experiment: The success rates ψ are shown in solid lines, the average time for a successful attempt t_ψ in dashed lines. The indexes stand for the stop criteria $N^F = 10^5$ and $N^F = 10^6$.

early stage of the simulation. This is caused by a few particles that get stuck and keep colliding with each other, for example near a corner. The larger specimen with $a = 80$ mm has about 16 times more particles, which increases the probability for these events. However, they are efficiently identified with the stopping criterion and barely influence the average computation time.

The particle distribution for $a = 80$ mm and $\phi_{\text{eff}} = 70\%$ is illustrated in Fig. 8. The RSA algorithm stopped after placing 3 166 of 9 367 particles, whereas the EDMD was successful.

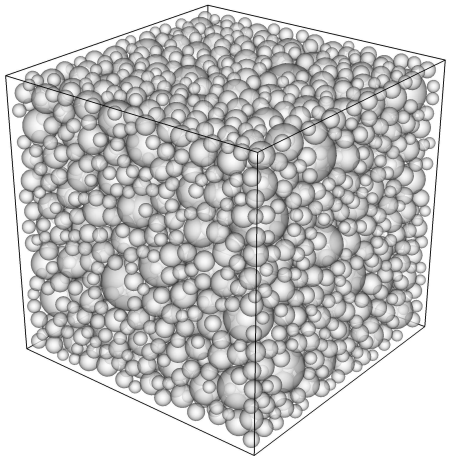
6.3. Meshable concrete specimen

In this numerical experiment, the particle size distribution of a real concrete specimens is simulated. The grading curve A16 according to DIN 1045-2 is chosen. As Section 3 describes, only particles larger than $d_{\min} = 2$ mm are considered and a distinction between ϕ_M and ϕ_{eff} is important. Additionally, a minimal distance of $\Delta d = d_{\min}/2 = 1$ mm between the particles is enforced by increasing their diameter by Δd during the particle placement.

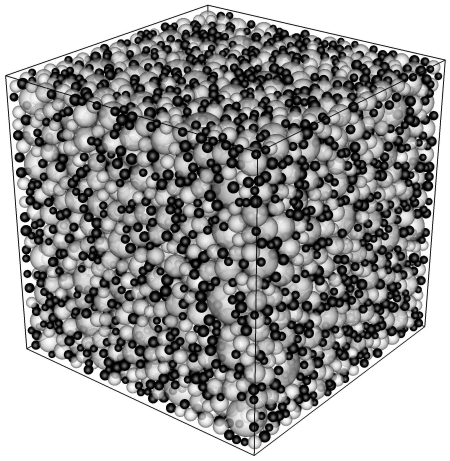
The RSA algorithm is performed with the modified diameters $d + \Delta d$. For the initial sphere distribution of the EDMD algorithm, the RSA algorithm with $\Delta d = 0$ is used. The absolute growth rate $\gamma_a = 0.1\,\mathrm{mm/s}$ is applied to each particle and the EDMD simulations runs until t_{end}. Since the growth rate γ applies to the particle radius and not its diameter, the factor 2 appears.

$$t_{\text{end}} = \frac{\Delta d}{2\gamma_a} \tag{31}$$

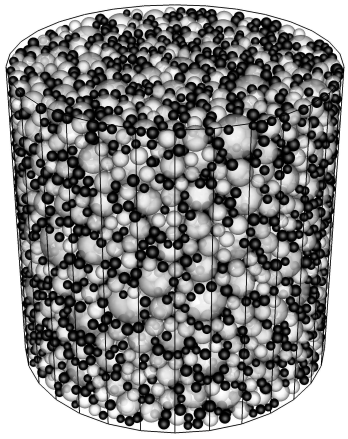
A cubic specimen with edge length $a = 5\ d_{\max}$ is used. Similar to the previous section, the volume fraction is increased in steps of $\Delta\phi_{\text{M}} = 0.5\%$, beginning at $\phi_{\text{M}} = 40\%$. In each step, 50 size distributions are simulated and the success rate ψ and the time t_ψ is calculated.



(a) RSA - unsuccessful attempt. The algorithm failed after placing the 3 166 largest of 9 367 total particles



(b) EDMD - successful attempt. The 3 166 biggest particles are colored white, the others black.



(c) EDMD - successful attempt for a cylindrical specimen with diameter a and height a.

Fig. 8: Result of the simulation with $a = 5\ d_{\max}$ and an effective volume fraction of $\phi_{\text{eff}} = 70\%$.

The results shown in Fig. 9 are qualitatively similar to Fig. 7. The smaller reachable particle volume is due to the minimum distance Δd between any two particles. The EDMD algorithm is able to create concrete structures with 11% more particle volume ($\phi_{\text{M,EDMD}} = 53.5\%$, $\phi_{\text{M,RSA}} = 48\%$). Since the smallest 21% of the particles mass are not explicitly modeled ($d < d_{\min}$), the corresponding effective volume fractions are $\phi_{\text{eff,EDMD}} = 42\%$ and $\phi_{\text{eff,RSA}} = 38\%$.

At $\phi_M = 53.5\%$, 90% of the EDMD runs were successful, the simulations had about $5\,000$ particles and took 135 s on average. About 4 million events occurred: 88% particle collisions, 10.5% wall collisions and 1.5% cell transfers (caused by the cell method). The additional volume due to Δd, i.e. the layer coating each particle with the thickness $\Delta d/2$, has a volume fraction of $\phi_{\Delta d} = 25\%$. With the effective volume fraction of the simulation, this adds up to a total volume fraction of 67%.

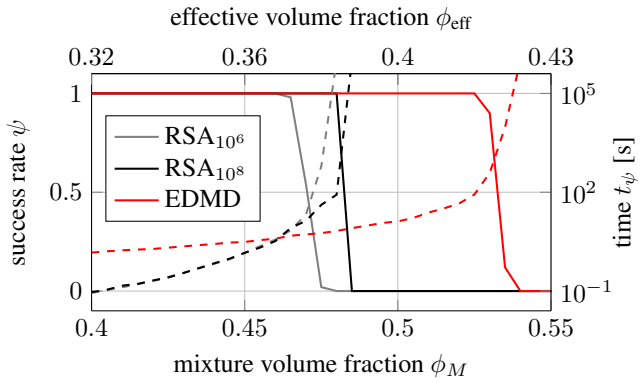


Fig. 9: Results of the meshable concrete specimen experiment with A16, $a = 80$ mm and a minimal distance $\Delta d = d_{\min}/2 = 1$ mm between the particles. The solid lines represent the success rates ψ, the dashed lines the time t_ψ. The indices stand for the stop criteria $N^F = 10^6$ and $N^F = 10^8$.

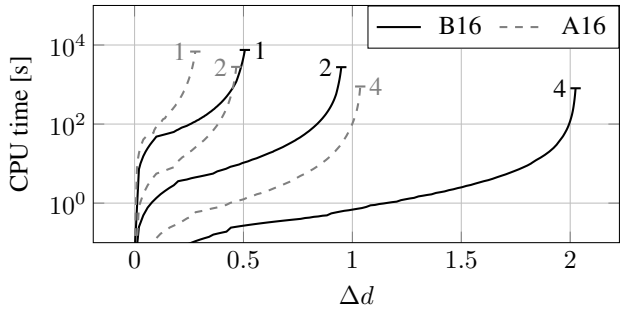


Fig. 10: CPU time needed to reach a certain Δd. The grading curves A16 and B16 with $\phi_M = 70\%$ are investigated for different cutoff diameters $d_{\min}$ (indicated by numbers [mm] in the graph).

6.4. Maximizing the particle distance

In the previous experiment and for the purpose of comparing the RSA and the EDMD algorithm, a fixed Δd is prescribed and ϕ_M is varied. In a more practical approach, the highest Δd for a given configuration is of interest. It can be connected directly to a minimal element length in finite element simulations and, thus, gives an approximation for the expected number of degrees of freedom.

Here, only the EDMD algorithm is used. Similar to the previous experiment, an initial size distribution that already matches the desired grading curve is obtained by the RSA algorithm. Now, an absolute growth rate γ_a is given to each particle. As the simulation time t passes, the distance Δd grows. The simulation is carried out until the stopping criterion ($\Delta t < 10^{-6} s$ in 1000 events) is reached.

A cubic specimen with an edge length $a = 150$ mm is used in all simulations. The grading curves A16 and B16, as well as the mixture volume fraction $\phi_M = 60\%$ and $\phi_M = 70\%$ are investigated. Additionally, the cutoff diameter $d_{\min}$ is varied.

For a specific set of parameters, the computation time needed to reach a certain Δd is shown in Fig. 10. When Δd reaches its maximum, the slope of the curves increases dramatically and the calculation stops.

Two phenomena can be analyzed in Fig. 10. First, with decreasing $d_{\min}$, the computation time increases. This is due to the growing number of simulated particles that causes more collisions and longer event lists. Second, B16 reaches a higher particle distance Δd compared to A16 - in this case about twice as much. Since B16 has a lower effective volume fraction than A16, it provides more space for the additional volume caused by Δd.

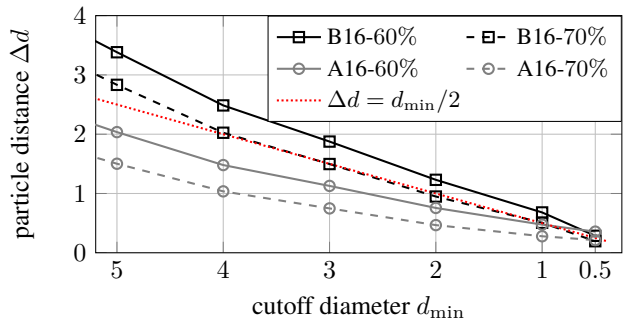


Fig. 11: Maximum Δd reachable at a certain cutoff diameter $d_{\min}$ for different grading curves and volume fractions. In the legend entries, the numbers indicate the particle volume fraction $\phi_M = 60\%$ and $\phi_M = 70\%$. The dotted red line represents the requirement $2\Delta d = d_{min}$ used in the previous sections.

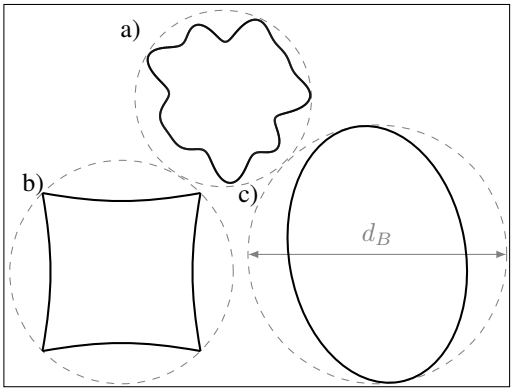


Fig. 12: The EDMD algorithm can be performed with bounding spheres of diameter d_B for each non-spherical particle. Arbitrary shapes e.g. a) based on spherical harmonics [33], b) deformed cuboids or c) ellipsoids can be used.

As shown in Fig. 11, the second effect occurs for different cutoff diameters d_{min} and different volume fractions. The B16 grading curve meets the requirements of $\Delta d = d_{\min}/2$ from the previous section for both volume fractions $\phi_M = 60\%$ and $\phi_M = 70\%$, whereas the A16 grading curve does not. The only exception is $d_{\min} = 0.5$ mm. Due to the grading curve design, the overall number of particles is about 2.05 million for B16 and only 0.82 million for A16 (see Fig. 3). Each particle is coated with a layer of thickness $\Delta d/2$. For a higher number of particles, this layer requires more volume. Consequently, the reachable Δd decreases. Since the number of particles grows with decreasing cutoff diameters, this dependence also explains the general decrease of Δd.

7. Non-spherical particles

Both algorithms in this paper use spherical particles. However, extensions to ellipsoidal particles are possible. A fast ellipsoid separation check by Wang et al. [30] allows an efficient handling of those particles in an RSA algorithm. An implementation is done by Häfner et al. [31]. In an EDMD algorithm for ellipsoids, their movement (including rotations), their collision behavior and their collision prediction must be considered. The latter one involves solving a quartic equation, in contrast to a quadratic one for spheres. This is efficiently implemented by Donev et al. [27, 32].

The shape of real concrete aggregates, e.g. obtained by x-ray tomography, is more complex. Garboczi used spherical harmonics to characterize arbitrarily shaped particles [33]. The functional description of the corresponding geometry is very complex and separation checks or collision predictions are difficult to implement. With a slight modification of the present EDMD algorithm these shapes can be represented as well. In the *take* phase, the non-spherical particles must be picked in accordance with the desired grading curve. Each particle is embedded in a bounding sphere whose diameter d_B matches the largest particle dimension. Their size distribution is then used as the input for the *place* phase with the EDMD algorithm. In a post-processing step, illustrated in Fig. 12, the non-spherical particles are placed in their bounding spheres.

However, with higher particle asphericity, the reachable volume fraction drops, since the bounding spheres are not completely filled. In this case, the previously described modifications using ellipsoids are advantageous.

8. Conclusions

In this paper, the RSA and the EDMD algorithm are used to create realistic concrete mesoscale geometries based on a given grading curve. Both methods are compared in terms of maximal reachable volume fraction and performance. For a fixed set of

parameters and varying volume fractions, the success rate and the average time for a successful geometry creation is measured.

In the RSA implementation, a cell method with variable cell sizes is used. At low volume fractions, it performs much faster than the EDMD algorithm. However, near its maximum volume fraction, its performance drops under the EDMD performance, until it is not able to create a valid geometry at all.

The present EDMD algorithm is a modified Lubachevski-Stillinger packing algorithm. A loosely packed initial state is obtained by the RSA method. Several optimization techniques, like the delayed state, the cell method, local event lists and time barriers, are implemented. The process of a single event scales logarithmically with the number of particles and dense geometries with over two million particles are created. Depending on the specific grading curve used, the maximum volume fraction of the algorithm exceeds the RSA value by up to 11%.

In real concrete specimens, a the minimal distance Δd between particles is observable. Maximizing this distance is advantageous for future finite element simulations. For this purpose, the presented EDMD algorithm is ideal. For the initial state a valid particle configuration matching the prescribed grading curve but without a minimal distance ($\Delta d = 0$) can be obtained either using RSA or EDMD. Using a constant growth rate for Δd, the particle distance is increased over time. A stopping criterion, e.g. the time increase for 1000 events, indicates a configuration with maximum particle distance Δd. The particles are shrunk back to their original size and now have a distance Δd between each other.

For the B16 grading curve and an aggregate volume fraction of 70%, the EDMD algorithm reaches a value of $\Delta d = d_{\min}/2$. This value is excellent for meshing the resulting geometry.

An extension of this algorithm to ellipsoidal particles is planned for future work. With the more sophisticated optimization techniques mentioned in this paper, the number of collision predictions can be further reduced and the complex ellipsoid movement and collision prediction becomes manageable.

Acknowledgment

The research was supported by the Federal Institute for Materials Research and Testing, Berlin, Germany and by the German Research Foundation (DFG) under project Un224/7-1. Additionally, the research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 320815 (ERC Advanced Grant Project "Advanced tools for computational design of engineering materials" COMP-DES-MAT).

References

[1] Skoge M, Donev A, Stillinger FH, Torquato S. Packing hyperspheres in high-dimensional Euclidean spaces. Physical Review E 2006;74:041127. doi:10.1103/PhysRevE.74.041127.

[2] Torquato S, Stillinger FH. Jammed hard-particle packings: From Kepler to Bernal and beyond. Reviews of Modern Physics 2010;82:2633–72. doi:10.1103/RevModPhys.82.2633.

[3] Farr RS. Random close packing fractions of lognormal distributions of hard spheres . Powder Technology 2013;245(0):28 – 34. doi:10.1016/j.powtec.2013.04.009.

[4] Hüsken G, Brouwers H. A new mix design concept for earth-moist concrete: A theoretical and experimental study. Cement and Concrete Research 2008;38(10):1246 –59. doi:10.1016/j.cemconres.2008.04.002.

[5] Nagai G, Yamada T. Three-Dimensional Finite Element Modeling for Concrete Materials Using Digital Image and Embedded Discontinuous Element. International Journal for Multiscale Computational Engineering 2006;4(4):461–74. doi:10.1615/IntJMultCompEng.v4.i4.40.

[6] Unger JF, Eckardt S, Könke C. A mesoscale model for concrete to simulate mechanical failure. Computers and Concrete 2011;8(4):401–23. doi:10.12989/cac.2011.8.4.401.

[7] Unger JF, Eckardt S. Multiscale Modeling of Concrete. Archives of Computational Methods in Engineering 2011;18(3):341–93. doi:10.1007/s11831-011-9063-8.

[8] Wang ZM, Kwan AKH, Chan HC. Mesoscopic study of concrete I: generation of random aggregate structure and finite element mesh . Computers & Structures 1999;70(5):533 –44. doi:10.1016/S0045-7949(98)00177-1.

[9] Wriggers P, Moftah SO. Mesoscale models for concrete: Homogenisation and damage behaviour. Finite Elements in Analysis and Design 2006;42(7):623 –36. doi:10.1016/j.finel.2005.11.008.

[10] Widom B. Random Sequential Addition of Hard Spheres to a Volume. The Journal of Chemical Physics 1966;44(10):3888–94. doi:10.1063/1.1726548.

[11] Leite JPB, Slowik V, Apel J. Computational model of mesoscopic structure of concrete for simulation of fracture processes . Computers & Structures 2007;85(17–18):1293 –303. doi:10.1016/j.compstruc.2006.08.086.

[12] Mier JGMV, Vliet MRAV. Influence of microstructure of concrete on size/scale effects in tensile fracture. Engineering Fracture Mechanics 2003;70(16):2281 –306. doi:10.1016/S0013-7944(02)00222-9.

[13] Alder B, Wainwright T. Studies in Molecular Dynamics. I. General Method. The Journal of Chemical Physics 1959;31(2):459–66. doi:10.1063/1.1730376.

[14] Lubachevsky BD, Stillinger FH. Geometric properties of random disk packings. Journal of Statistical Physics 1990;60(5-6):561–83. doi:10.1007/BF01025983.

[15] Kansal AR, Torquato S, Stillinger FH. Computer generation of dense polydisperse sphere packings. The Journal of Chemical Physics 2002;117(18):8212–8. doi:10.1063/1.1511510.

[16] Kansal AR, Torquato S, Stillinger FH. Diversity of order and densities in jammed hard-particle packings. Physical Review E 2002;66(4):041109. doi:10.1103/PhysRevE.66.041109.

[17] Sirmas N, Tudorache M, Barahona J, Radulescu MI. Shock waves in dense hard disk fluids. Shock Waves 2012;22(3):237–47. doi:10.1007/s00193-012-0354-2.

[18] Bannerman M, Magee J, Lue L. Structure and stability of helices in square-well homopolymers. Physical Review E 2009;80:021801. doi:10.1103/PhysRevE.80.021801.

[19] Mandal S, Gross M, Raabe D, Varnik F. Heterogeneous Shear in Hard Sphere Glasses. Physical Review Letteers 2012;108:098301. doi:10.1103/PhysRevLett.108.098301.

[20] Sonon B, François B, Massart TJ. A unified level set based methodology for fast generation of complex microstructural multi-phase RVEs . Computer Methods in Applied Mechanics and Engineering 2012;223–224(0):103 –22. doi:10.1016/j.cma.2012.02.018.

[21] Jerier JF, Imbault D, Donze FV, Doremus P. A geometric algorithm based on tetrahedral meshes to generate a dense polydisperse sphere packing. Granular Matter 2009;11(1):43–52. doi:10.1007/s10035-008-0116-0.

[22] Cundall PA, Strack OD. A discrete numerical model for granular assemblies. Geotechnique 1979;29(1):47–65. doi:10.1680/geot.1979.29.1.47.

[23] Fu G, Dekelbab W. 3-D random packing of polydisperse particles and concrete aggregate grading . Powder Technology 2003;133(1–3):147 –55. doi:10.1016/S0032-5910(03)00082-2.

[24] Wittmann FH, Roelfstra PE, Sadouki H. Simulation and analysis of composite structures . Materials Science and Engineering 1985;68(2):239 –48. doi:10.1016/0025-5416(85)90413-6.

[25] Schlangen E, Van Mier JGM. Simple lattice model for numerical simulation of fracture of concrete materials and structures. Materials and Structures 1992;25(9):534–42. doi:10.1007/BF02472449.

[26] Lubachevsky BD. How to simulate billiards and similar systems . Journal of Computational Physics 1991;94(2):255 –83. doi:10.1016/0021-9991(91)90222-7.

[27] Donev A, Torquato S, Stillinger FH. Neighbor list collision-driven molecular dynamics simulation for nonspherical hard particles. I. Algorithmic details . Journal of Computational Physics 2005;202(2):737 –64. doi:10.1016/j.jcp.2004.08.014.

[28] Paul G. A Complexity $\mathcal{O}(1)$ priority queue for event driven molecular dynamics simulations. Journal of Computational Physics 2007;221(2):615 –25. doi:10.1016/j.jcp.2006.06.042.

[29] Bannerman M, Sargant R, Lue L. DynamO: a free $\mathcal{O}$(N) general event-driven molecular dynamics simulator. Journal of Computational Chemistry 2011;32(15):3329–38. doi:10.1002/jcc.21915.

[30] Wang W, Wang J, Kim MS. An algebraic condition for the separation of two ellipsoids . Computer Aided Geometric Design 2001;18(6):531 –9. doi:10.1016/S0167-8396(01)00049-8.

[31] Häfner S, Eckardt S, Luther T, Könke C. Mesoscale modeling of concrete: Geometry and numerics . Computers & Structures 2006;84(7):450 –61. doi:10.1016/j.compstruc.2005.10.003.

[32] Donev A, Torquato S, Stillinger FH. Neighbor list collision-driven molecular dynamics simulation for nonspherical hard particles.: II. Applications to ellipses and ellipsoids. Journal of Computational Physics 2005;202(2):765–93. doi:10.1016/j.jcp.2004.08.025.

[33] Garboczi EJ. Three-dimensional mathematical analysis of particle shape using x-ray tomography and spherical harmonics: Application to aggregates used in concrete. Cement and concrete research 2002;32(10):1621–38. doi:10.1016/S0008-8846(02)00836-0.