16th World Congress in Computational Mechanics (WCCM)
21-26 July 2024, Vancouver, Canada
A. Korobenko, M. Laforest, S. Prudhomme, R. Vaziri (Eds)

PYMESOSCALE: A FRIENDLY PYTHON LIBRARY FOR GENERATING 3D CONCRETE MESOSCALE MODELS

FAISAL L. MUHAMMAD[1] AND SAHEED K. ADEKUNLE[2]

[1] Department of Civil & Environmental Engineering, King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia.
g202210460@kfupm.edu.sa

[2] Department of Civil & Environmental Engineering, King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia;
Interdisciplinary Research Center for Construction and Building Materials, King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia.
saheedka@kfupm.edu.sa; orcid.org/0000-0003-2160-1502

Keywords: Concrete, Aggregate, Mesoscale model, Interfacial transition zone, Python.

Abstract. Mesoscale modeling of concrete and its composites has attracted a lot of researchers over the years with the promise of establishing an accurate relationship between the mesoscopic model and the macroscopic mechanical properties of concrete. The primary constraints inhibiting the widespread application of mesoscale modeling include (i) achieving high packing density with control over aggregate shape and gradation, (ii) high computational cost, (iii) accomplishing realistic interfacial transition zone, and (iv) implementing efficient aggregate intrusion detection. Besides these constraints, one major challenge is the lack of open program codes for algorithms implemented in published research papers. For example, improving, upgrading, and repurposing an existing algorithm by other researchers requires the open availability of the codes. The unavailability of these codes and the common absence of subtle implementation details in published papers hinder research progress on mesoscale modeling of concrete composites. This paper presents pyMesoscale, a Python library for generating 3D mesoscale models for concrete-like composites. pyMesoscale implements the local background method, a highly effective mesoscale generation algorithm that offers a much better aggregate intrusion detection system and a high aggregate volume fraction. Developed with Python due to its smoother learning curve and beginner friendliness, pyMesoscale reduces implementation complexity for ease of use by both new and experienced researchers in the field. This paper offers the mathematical formulars and the detailed steps required to implement the generation of mesoscale geometric model of concrete and its derived mesoscale finite element model covering aggregate gradation and shape, random aggregate translation and aggregate intrusion detection.

1.0 INTRODUCTION

Concrete, characterized by significant heterogeneities, is extensively utilized globally as a construction material and serves numerous purposes in infrastructural development. Damage and failure processes in concrete are closely related to its material composition and the interactions among its various phases. Typically, macroscopic damage begins with fractures along the interfaces between aggregates and the mortar matrix, known as the interfacial transition zone (ITZ). The degree of heterogeneity required in a computational model is influenced by the observation scale and the specific stress-strain conditions, especially the gradients involved [1].

Concrete can be modeled and analyzed as a multiscale material, incorporating macroscale, mesoscale, and microscale perspectives, as illustrated in Fig. 1 [2]. While concrete appears homogeneous at the macroscale, it is heterogeneous internally. Mesoscale models are extensively used to understand concrete's mechanical properties and failure mechanisms, as well as the contributions of various phases to its overall behavior. These models provide accurate homogenized responses at the macroscale by accounting for heterogeneous properties. Mesoscale modeling offers a distinctive approach to studying the initiation and coalescence of microcracks into larger cracks that lead to concrete failure [3], and captures localized deformations that homogenous continuum models miss. To fully grasp concrete's failure process, its inherent heterogeneity must be considered, with mesoscale modeling being the most effective method for understanding fracture behavior due to its ability to capture these heterogeneities [4,5]. Macroscale simulations are limited by inadequate material models for representing concrete's nonlinear behavior when its constituent material changes. However, mesoscale models can provide homogenized responses for different mix proportions by applying known material models to their components [6]. At the mesoscopic level, concrete is viewed as a composite material comprising three phases: ITZ, coarse aggregates, and mortar matrix. The ITZ properties differ significantly from the mortar matrix due to porosity gradients and anhydrous cement variations [7], making it crucial to model these phases separately in mesoscale models [8].

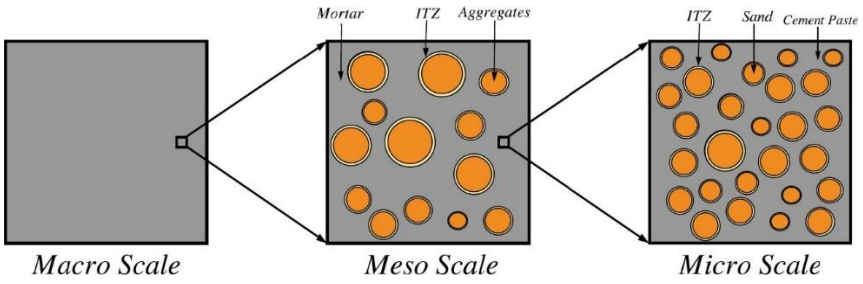


Figure 1 Multiple concrete scales [9].

Mesoscale models are essential for identifying factors that limit the performance of concrete mixes, such as determining if a weak phase at the mesoscale (coarse aggregate, mortar, or ITZ) controls the concrete's strength. Mesoscale modeling is applied to study multiple phenomena such as mechanical responses, hydration processes [10,11], thermomechanical behavior under fire conditions [12,13], cement-hydration interactions [14], wave propagation [15,16], dynamic loading responses [17–19], cracking caused by reinforcement corrosion [20], moisture transport [21], and diffusion of chloride in concrete [22].

In general, developing a concrete meso-model involves two main steps: (i) creating the meso-geometric model of concrete and (ii) meshing this geometric model to generate the mesoscale finite element model. Recent literature indicates that there are three primary methods for generating meso-geometric models of concrete: (i) the random aggregate placement method [27–31], (ii) Voronoi graphics method [32–36], and (iii) the XCT scanning method [37,38].

The random aggregate placement, or 'take-and-place', method involves randomly positioning pre-generated aggregates into the specimen's target geometric area to form a mesoscale geometric model of concrete. This approach effectively meets aggregate shape, size, and gradation distribution requirements. However, the random nature of the placement processes and the extensive intrusion detection needed for aggregates make the take-and-place procedure quite time-consuming, posing challenges for achieving high aggregate volume fraction or content [29]. Various strategies have been employed to address these issues to increase the volume fraction of aggregates. For example, the 'occupation and removal method' [29] improves placement efficiency, while the 'supplementary aggregates creation' directly boosts aggregate content [36], as illustrated in Fig. 2 (a) and (b).

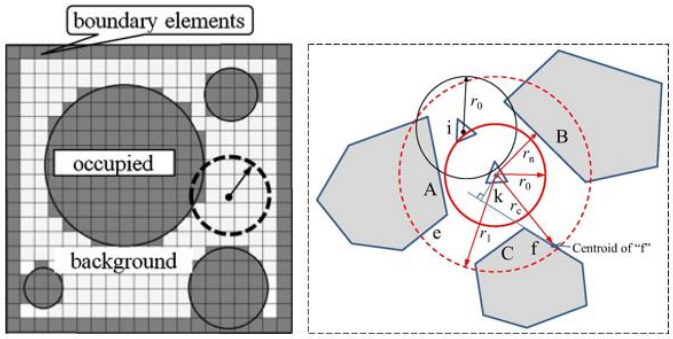


Figure 2 (a) Diagram of the occupation and removal technique, (b) Diagram of virtual sphere placement and intersection verification [29,39].

The Voronoi diagram method utilizes a three-dimensional Voronoi technique to partition the geometric region of concrete specimens into adjacent polyhedrons. By 'shrinking' these polyhedrons, isolated polyhedrons representing concrete aggregates are formed. This method allows for high aggregate content and realistic aggregate shapes, but it struggles to meet specific aggregate gradation distributions. Typically, a two-phase mesoscale geometric model created with the Voronoi method, which includes only aggregates and mortar, can achieve an aggregate content of up to 80% [36,40]. However, including the ITZ for a three-phase model decreases the aggregate content to around 70%.

This paper introduces pyMesoscale, a handy tool for mesoscale modeling of concrete. pyMesoscale implements the local background method for generating 3D mesoscale models, one of the most effective mesoscale generation algorithms, particularly with a much better aggregate intrusion detection during aggregate placement and a high aggregate volume fraction after packing. The generated aggregates are randomly translated onto a global mesh where all the elements have been initialized as mortar. A local background grid is generated around the translated aggregate where all aggregate and ITZ identification and intrusion occur.

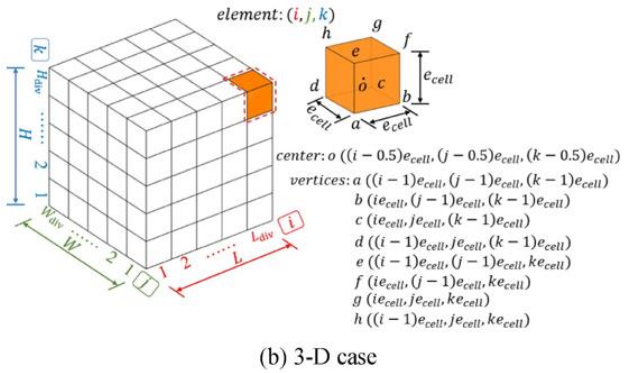
The pyMesoscale library was written in the Python programming language due to its numerous open-source libraries and active developer environment [23,24]. Two features are especially important for our purposes: (i) the clear syntax and simple semantics help new researchers against potential distractions inherent in complex syntaxes, allowing them to focus on the core programming concepts, and (ii) the rich collection of scientific libraries and toolkits for advanced tasks available for Python due to its widespread usage [25,26].

The mathematical formulation and detailed implementation methodology leading to the development of pyMesoscale are highlighted in this paper. Section 2.0 gives an overview of these mathematical formulars for generating mesoscale models for concrete using the local background grid method expatiating on the aggregate gradation and geometry as well as the random aggregate placement, material attribute identification and checks for possible aggregate intrusion. The final section shows how to install the pyMesoscale library from the python package index (PYPI) as well a basic demonstration on its usage to generate mesoscale models for visualization in ParaView and analysis in Abaqus.

2.0 METHODOLOGY

Using the background grid method to divide the finite element mesh elements ensures high quality and uniform size, which is advantageous for finite element calculations[41] Initially, the dimensions of the concrete sample need specification (for 2-D: 3-D) along with the element size. Firstly, we determine the size of the concrete sample which is given as $L \times W \times H$ since we are dealing with 3-D model. It's crucial that the element size is divisible by each side of the concrete sample. Choosing a reasonable element size is important; a large size can result in inaccurate display of aggregate outlines, while a small size increases the number of elements and calculation time[42]. According to Ouyang et al [41], we should take the mesh element size e to fall within the range of $\frac{1}{4} \sim \frac{1}{8}$ of the smallest size of aggregate.

Thus, the total number of elements or grids in the model can be calculated as $N_{elem} = L_{div} \times W_{div} \times H_{div}$. Where $L_{div} = L/e$, $W_{div} = W/e$, and $H_{div} = H/e$. In the 3D scenario, i, j and k represent the sequence numbers of the elements in the length, width, and height directions, respectively. Detailed coordinate calculations for the eight vertices and the center point of the element are shown in Fig. 4 below.



(b) 3-D case

Figure 3 Diagram illustrating background grid calculation calculation [42].

The central point coordinates for every element are as follows:

$$(ie + \frac{e}{2}, je + \frac{e}{2}, ke + \frac{e}{2}) \quad (1)$$

As a result, the number of the element where a point (x, y, z), is placed can also be found as follows if the coordinates of any point in the background mesh block are:

$$(x//e, y//e, z//e) \quad (2)$$

All the global background element's material qualities are kept in a three-dimensional matrix $M_{ijk}(0 \leq i \leq l-1, 0 \leq j \leq m-1, 0 \leq k \leq n-1)$. The grid element in this paper is either an aggregate element, an ITZ element, or a mortar element, depending on if $M_{ijk} = 1,2, or\ 3$.

2.1 Gradation of Aggregate

Two factors make up the aggregate gradation: (1) the aggregate particle size; and (2) the particle size distribution, or the quantity of aggregate of each size. Particles larger than 4.75mm in size are referred to be coarse aggregate for regular concrete, and they make up between 40 and 50 percent of the volume of the material [41]. A specific content of aggregates with varying particle sizes makes up the optimal gradation [11]. The constructed concrete will achieve stronger strength and improved compactness if the contents of each aggregate size are near the optimum continuous gradation curve. This paper uses the Fuller gradation curve, as indicated by Eq. 1 [9].

$$P(d) = \left(\frac{d}{D_{max}}\right)^{m} \times 100\% \quad (3)$$

Where D_{max} is the maximum aggregate particle size and $P(d)$ is the cumulative volume percentage of aggregates having a particle size less than d. Fig. 1 displays the aggregate gradation curve for concrete. Generally, speaking, the empirical exponent n ranges from 0.45 to 0.70 according to [42]. In another account, it ranges from 0.3 to 0.5, when mesoscale modelling was conducted for epoxy [43]. Nevertheless, user-defined aggregate gradation distribution will be supported in future updates of our library.

2.2 Aggregate shape

The shape and roughness of the particles affect the mechanical properties of concrete [44,45]. Compared to rounded particles, angular particles increase the strength of concrete due to its higher surface area, internal friction, and better mechanical interlock [46,47]. Aggregates have been represented in two dimensions using planar shapes such as spherical, ellipsoidal, and polyhedral shapes [48]. Because it makes creating particle shapes easier, spherical shapes are frequently utilized for aggregates [49–51]. The radius of the particles and their center coordinates provide a unique 3D representation of spheres.

2.2.1 Polyhedral Aggregate

Here we provide an implementation for only polyhedral aggregate as ellipsoidal and spherical aggregates are simple to implement [9]. The polyhedron's vertices are determined by choosing points at random from the auxiliary sphere using spherical coordinates. This study limits the number of polyhedron vertices to between 15 and 25, consistent with references [11,39,41]. For aggregates of a certain particle size, the auxiliary sphere radius is computed using Eq. (4) and (5).

$$\begin{cases}\theta_i = \eta \times 2\pi \\ \varphi_i = \xi \times 2\pi\end{cases} \tag{4}$$

$$\begin{cases}x_i = r \times \sin\varphi_i \times \cos\theta_i \\ y_i = r \times \sin\varphi_i \times \sin\theta_i \\ z_i = r \times \cos\theta_i\end{cases} \tag{5}$$

Let u and v be random variables within the range $[0, 1]$, where θ and φ represent the azimuthal and zenith angles in spherical coordinates. The entire set of polyhedron vertices undergoes translation according to Eq. 7. The geometric topology of the vertices is essential for defining the polyhedral aggregate's shape.

2.3 Random placement of aggregate and the local background grid

Element attribute recognition and aggregate intrusion detection are carried out in the local background grid, a dynamic bounding box associated with the position and form of the current aggregate [42].

The randomly generated polyhedral (or spherical, ellipsoidal) aggregate is inserted into the background grid space through random translation. Assume the coordinates of the N vertices of the polyhedral aggregates are (x_i, y_i, z_i), and the coordinates of the random translation point P are $(x_p, y_p, z_p) = (\eta_7 le,\ \eta_8 le, \eta_9 le)$, with η_7, η_8, and η_9 being random numbers within the range $[0,1]$. As a result, the coordinates for all vertices of the aggregate after applying the random translation are:

$$X_i = x_i + x_p, \quad Y_i = y_i + y_p, \quad Z_i = z_i + z_p \ (0 \le i \le N-1) \tag{6}$$

The boundary conditions for randomly placing aggregates may differ based on the preparation method of the concrete specimens. It is essential to ensure that the ransom translation point P falls within the confines of the concrete specimen when a block is extracted from the whole specimen. All vertices $(X_i, Y_i, and\ Z_i)$ of the randomly placed aggregate must be appropriately controlled to remain within the designated area of the concrete specimen.

$$0 \le X_i \le le,\ 0 \le Y_i \le me,\ 0 \le Z_i \le ne \quad (0 \le i \le N-1) \tag{7}$$

The random translation point P can be adjusted to ensure it is located within the mortar elements, thereby enhancing the efficiency of random aggerate placement. This means that the element containing the random translation point P should have a material attribute value of 1. The lowest values $(X_{min}, Y_{min}, Z_{min})$ and maximum values $(X_{max}, Y_{max}, Z_{max})$ of all the vertex coordinates in the three-dimensional direction are derived based on the new coordinates of the vertices of the polyhedral aggregate by Eq. 9.

$$\begin{aligned}(i_r, j_r, k_r) &= (X_{min}//e,\ Y_{min}//e,\ Z_{min}//e) \\ (I_r, J_r, K_r) &= (X_{max}//e,\ Y_{max}//e,\ Z_{max}//e)\end{aligned} \tag{8}$$

Some of the polyhedral aggregate's vertices may be outside the concrete block when looking at a concrete block that was cut from an entire concrete specimen. Therefore, Eq. 9. Should be used to find the bounding box of the newly put aggregate.

$$(i_r, j_r, k_r) = (\max(0, X_{min}//e)),\ \ \max(0, Y_{min}//e),\ \ \max\ (0,\ Z_{min}//e)$$
$$(I_r, J_r, K_r) = (\min(l, X_{max}//e)),\ \ \min(m, Y_{max}//e),\ \ \max\ (\mathrm{n},\ Z_{max}//e) \quad (9)$$

When numerically modeling concrete, the interfacial transition zone (ITZ) between the mortar and aggregate must be considered due to its significant impact on the material's mechanical properties. The real ITZ is quite thin, typically between 10 to 50 micrometers. Due to computational limitations, the mesh element size in numerical models only roughly matches the actual ITZ size. In this study, the ITZ is created by designating a layer/unit of element cells around the aggregate's outer contour as ITZ element. Consequently, an additional layer accounting for the ITZ elements must be incorporated into the bounding box as illustrated in Fig. 5 (a).

$$(i_b, j_b, k_b) = (\max(0, X_{min}//e - 1)),\ \ \max(0, Y_{min}//e - 1),\ \ \max\ (0,\ Z_{min}//e - 1)$$
$$(I_b, J_b, K_b) = (\min(l, X_{max}//e + 1)),\ \ \min(m, Y_{max}//e + 1),\ \ \max\ (\mathrm{n},\ Z_{max}//e + 1) \quad (10)$$

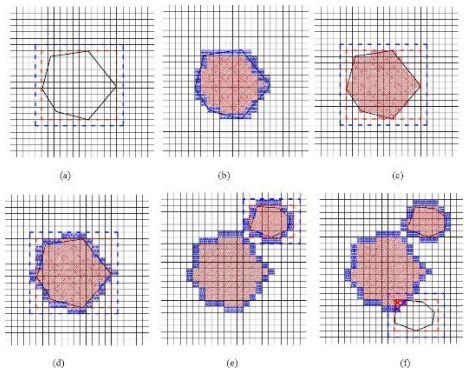


Figure 4 (a) random placement of aggregates with the local background grid. (b) traditional identification method. (c) identification of aggregate elements. (d) creation of ITZ elements. (e) addition of a new aggregate. (f) overlap of new and existing aggregates [42].

2.4 Identification of material attributes and detection of aggregate intrusion

A 3D dynamic matrix denoted as B, is utilized to temporarily hold the material properties of the current local background grid. Matrix B is initialized with a value of 1, indicating that all elements within the local grid are initially set as mortar components. The relationship between elements in the local background grid matrix B and those in the global background Matrix M is described as follows:

$$\begin{aligned} M & \quad \Leftrightarrow \quad B \\ (i_b, j_b, k_b) & \quad \Leftrightarrow \quad (0,0,0) \\ (i + i_b, j + j_b, k + k_b) & \quad \Leftrightarrow \quad (i, j, k) \\ (I, J, K) & \quad \Leftrightarrow \quad (I - i_b, J - j_b, K - k_b) \\ (I_b, J_b, K_b) & \quad \Leftrightarrow \quad (I_b - i_b, J_b - j_b, K_b - k_b) \end{aligned} \tag{11}$$

The material properties of all elements within the local background grid are determined based on their positional relationship to the aggregate geometry. According to various sources [28,29], the traditional method for classifying element materials is as follows:

1. An element is classified as an aggregate element if all its nodes lie within the aggregate geometry.
2. An element is classified as a mortar element if all its nodes lie outside the aggregate geometry.
3. If the element's nodes do not fit entirely inside or outside the aggregate, it is classified as an ITZ element.

3.0 CONCLUSION

To access the python library, we need to download it from the python package index (PYPI), the index provides a way of easily accessing open-source python code written by other programmers to automate and bring structure to some repetitive tasks when programming. To install the library, we need to have python installed on our system/laptop and have a development environment usually an Interactive development environment (IDE). An example of such an environment is the visual studio code usually pronounced as *"vs code"*. Afterwards we run the *pip install pymesoscale* command in our terminal provided either inside *vs code* or our laptop.

Table 1 Installing python library from PYPI

```
pip install pymesoscale
```

This command written in the computer's terminal downloads the latest version of the library. With the library installed we can now use it to generate mesoscale models. These models will be provided in the form of a *.vti* file format, which is used to store series of image data and allows visualization using software like *Paraview*. Apart from visualization, the library provides you with your model in a *.inp* file format. This format is known to belong to the Abaqus CAE (Computer Aided Engineering) Simulation software. We can import our models into Abaqus using the "import parts" option. This was a major solution as most of the available mesoscale generation code encountered in the *GitHub* online repository provides no means of doing so. Table. 2 shows the usage of the library to generate a mesoscale model in python.

Table 2 Utilizing the library to generate a mesoscale model of concrete.

```
from pymesoscale.local_background_grid.mesoscale import Mesoscale

# Trying to generate and place aggregates.
```

```
m = Mesoscale(100, 100, 100, 5, 20, 5000)
# export 3D array to vti file for visualization with Paraview
m._export_data(m.glob, export_type="vtk", fileName="mesoscale_model_16.vti")
# convert vti file -> vtu file -> .inp file for analysis in abaqus.
m.convert_vti_to_inp("mesoscale_model_16.vti", "output_topology_16")
```

Afterwards, we can visualize our concrete mesoscale model using the ParaView software as illustrated in Fig. 7. Below.

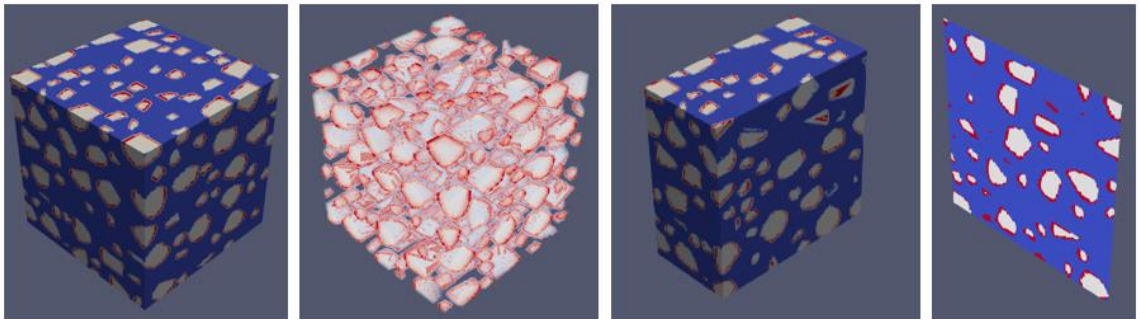


Figure 5 (a) Generated mesoscale model where blue is the mortar, white is the aggregate and red is the ITZ (b) Volumetric representation of the model (c) Clip view of the model (d) Slice View of the model.

The *convert_vti_to_inp* function is responsible for converting the generated mesoscale model in *.vti* or *.vtu* format into a *.inp* file format for easy importing into a Computer Aided Engineering (CAE) commercial software like Abaqus for modelling and simulation. The library splits the model into separate *.inp* part for easy assignment of material properties as illustrated in Fig. 9.

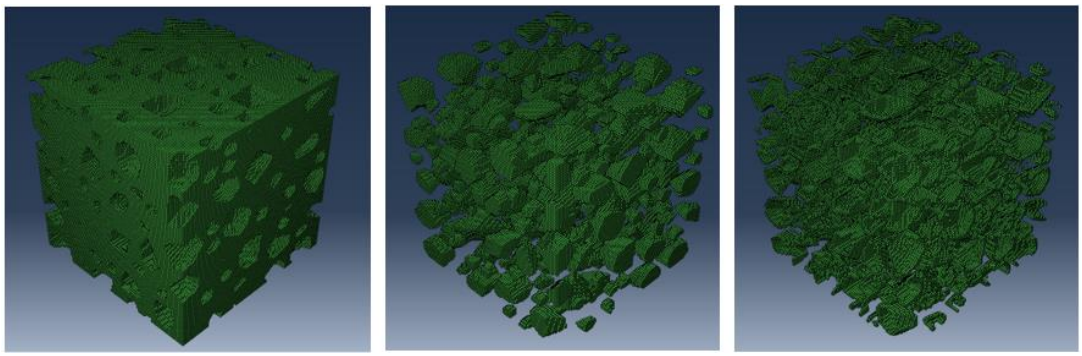


Figure 6 (a) generated mesoscale mortar model converted to inp for Abaqus simulation (b) Aggregate model converted to inp (c) ITZ model converted to inp.

4.0 CLOSING REMARKS

We set out to develop an open-source python library called pyMesoscale that solves the problem of generating mesoscale models for concrete enabling research in fracture mechanics of concrete specimens. Below are some of the highlights of our library.

1. Easy and fast generation of mesoscale model of concrete using the local background grid method with high aggregate volume fraction of polygonal aggregates requiring only the basic knowledge and little experience of python programming.

2. We also provide a way of exporting the model to a format suitable for simulation using commercial computer aided engineering (CAE) software such as Abaqus. Our output file is in the form of a *vtk* file allowing visualization with ParaView and a *.inp* file for use in Abaqus Simulations.
3. Future updates to our library will include various types of aggregate geometry such as spherical and ellipsoidal. Various types of concrete specimen shape, such as cylindrical. 2D mesoscale modelling, incorporation of recycled aggregates as well as other mesoscale model generation algorithms.

5.0 RECOMMENDATIONS

There is still a long way to go to turn the library into a mature toolkit that enables researchers to conduct computational fracture mechanics using mesoscale modelling of various concrete geometries, aggregate shapes, packing density, recycled aggregates, composites etc. I invite researchers from other universities to contribute to the repo of the project on GitHub. It will require a monumental effort beyond the capability of one person to reach such a goal. A goal that might extend into other research domains beyond mesoscale modelling of concrete.

REFERENCES

[1] Zhou R, Song Z, Lu Y. 3D mesoscale finite element modelling of concrete. Comput Struct 2017;192:96–113. https://doi.org/10.1016/J.COMPSTRUC.2017.07.009.

[2] Nguyen VP, Stroeven M, Sluys LJ. Multiscale failure modeling of concrete: Micromechanical modeling, discontinuous homogenization and parallel computations. Comput Methods Appl Mech Eng 2012;201–204:139–56. https://doi.org/10.1016/j.cma.2011.09.014.

[3] Guo H, Ooi ET, Saputra AA, Yang Z, Natarajan S, Ooi EH, et al. A quadtree-polygon-based scaled boundary finite element method for image-based mesoscale fracture modelling in concrete. Eng Fract Mech 2019;211:420–41. https://doi.org/10.1016/j.engfracmech.2019.02.021.

[4] Daoud A, Maurel O, Laborderie C. 2D mesoscopic modelling of bar-concrete bond. Eng Struct 2013;49:696–706. https://doi.org/10.1016/j.engstruct.2012.11.018.

[5] Wu M, Zhang C, Chen Z. Determining the impact behavior of concrete beams through experimental testing and meso-scale simulation: II. Particle element simulation and comparison. Eng Fract Mech 2015;135:113–25. https://doi.org/10.1016/j.engfracmech.2014.12.020.

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

[7] Ollivier JP, Maso JC, Bourdette B. Interfacial transition zone in concrete. Advanced Cement Based Materials 1995;2:30–8. https://doi.org/10.1016/1065-7355(95)90037-3.

[8] Zheng JJ, Guo ZQ, Huang XF, Stroeven P, Sluys LJ. ITZ volume fraction in concrete with spheroidal aggregate particles and application: Part II. Prediction of the chloride diffusivity of concrete. Magazine of Concrete Research 2011;63:483–91. https://doi.org/10.1680/macr.2011.63.7.483.

[9] Thilakarathna PSM, Kristombu Baduge KS, Mendis P, Vimonsatit V, Lee H. Mesoscale modelling of concrete – A review of geometry generation, placing algorithms, constitutive relations and applications. Eng Fract Mech 2020;231. https://doi.org/10.1016/j.engfracmech.2020.106974.

[10] Hou D, Zhang W, Wang P, Wang M, Zhang H. Mesoscale insights on the structure, mechanical performances and the damage process of calcium-silicate-hydrate. Constr Build Mater 2021;287:123031. https://doi.org/10.1016/j.conbuildmat.2021.123031.

[11] Zheng Z, Wei X. Mesoscopic models and numerical simulations of the temperature field and hydration degree in early-age concrete. Constr Build Mater 2021;266:121001. https://doi.org/10.1016/j.conbuildmat.2020.121001.

[12] Jin L, Zhang R, Li L, Du X, Yao Y. Impact behavior of SFRC beams at elevated temperatures: Experimental and analytical studies. Eng Struct 2019;197:109401. https://doi.org/10.1016/j.engstruct.2019.109401.

[13] Nguyen TD, Pham DT, Vu MN. Thermo-mechanically-induced thermal conductivity change and its effect on the behaviour of concrete. Constr Build Mater 2019;198:98–105. https://doi.org/10.1016/j.conbuildmat.2018.11.146.

[14] Homel MA, Iyer J, Semnani SJ, Herbold EB. Mesoscale model and X-ray computed micro-tomographic imaging of damage progression in ultra-high-performance concrete. Cem Concr Res 2022;157:106799. https://doi.org/10.1016/J.CEMCONRES.2022.106799.

[15] Chen C, Bai S, Huang Y, Lam L, Yao Y, Keer LM. 3D random packing algorithm of ellipsoidal particles based on the Monte Carlo method. Https://DoiOrg/101680/Jmacr2000228 2021;73:343–55. https://doi.org/10.1680/JMACR.20.00228.

[16] Jin H, Hao H, Chen W, Xu C. Spall Behaviors of Metaconcrete: 3D Meso-Scale Modelling. International Journal of Structural Stability and Dynamics 2021;21:2150121. https://doi.org/10.1142/S0219455421501212.

[17] Guo R, Ren H, Zhang L, Long Z, Jiang X, Wu X, et al. Direct dynamic tensile study of concrete materials based on mesoscale model. Int J Impact Eng 2020;143:103598. https://doi.org/10.1016/j.ijimpeng.2020.103598.

[18] Saksala T. Numerical modelling of concrete fracture processes under dynamic loading: Meso-mechanical approach based on embedded discontinuity finite elements. Eng Fract Mech 2018;201:282–97. https://doi.org/10.1016/j.engfracmech.2018.07.019.

[19] Tang L, Zhou W, Liu X, Ma G, Chen M. Three-dimensional mesoscopic simulation of the dynamic tensile fracture of concrete. Eng Fract Mech 2019;211:269–81. https://doi.org/10.1016/j.engfracmech.2019.02.015.

[20] Du X, Jin L, Zhang R. Modeling the cracking of cover concrete due to non-uniform corrosion of reinforcement. Corros Sci 2014;89:189–202. https://doi.org/10.1016/J.CORSCI.2014.08.025.

[21] Caggiano A, Said Schicchi D, Mankel C, Ukrainczyk N, Koenders EAB. A mesoscale approach for modeling capillary water absorption and transport phenomena in cementitious materials. Comput Struct 2018;200:1–10. https://doi.org/10.1016/j.compstruc.2018.01.013.

[22] Chen X, Yu A, Liu G, Chen P, Liang Q. A multi-phase mesoscopic simulation model for the diffusion of chloride in concrete under freeze–thaw cycles. Constr Build Mater 2020;265:120223. https://doi.org/10.1016/j.conbuildmat.2020.120223.

[23] Introduction to Computing: Explorations in Language, Logic, and Machines n.d. https://computingbook.org/ (accessed May 24, 2024).

[24] Fourment M, Gillings MR. A comparison of common programming languages used in bioinformatics. BMC Bioinformatics 2008;9:1–9. https://doi.org/10.1186/1471-2105-9-82/TABLES/1.

[25] Cieślik M, Mura C. A lightweight, flow-based toolkit for parallel and distributed bioinformatics pipelines. BMC Bioinformatics 2011;12:1–11. https://doi.org/10.1186/1471-2105-12-61/FIGURES/5.

[26] Hinsen K. The Molecular Modeling Toolkit: A New Approach to Molecular Simulations. J Comput Chem 2000;21:79–85. https://doi.org/10.1002/(SICI)1096-987X(20000130)21:2.

[27] Zhang J, Wang Z, Yang H, Wang Z, Shu X. 3D meso-scale modeling of reinforcement concrete with high volume fraction of randomly distributed aggregates. Constr Build Mater 2018;164:350–61. https://doi.org/10.1016/J.CONBUILDMAT.2017.12.229.

[28] Approach to generation of random convex polyhedral aggregate model and plotting for concrete meso-mechanics n.d. https://www.researchgate.net/publication/292649700_Approach_to_generation_of_random_convex_polyhedral_aggregate_model_and_plotting_for_concrete_meso-mechanics (accessed May 20, 2024).

[29] Ma H, Xu W, Li Y. Random aggregate model for mesoscopic structures and mechanical analysis of fully-graded concrete. Comput Struct 2016;177:103–13. https://doi.org/10.1016/J.COMPSTRUC.2016.09.005.

[30] Wang X, Zhang M, Jivkov AP. Computational technology for analysis of 3D meso-structure effects on damage and failure of concrete. Int J Solids Struct 2016;80:310–33. https://doi.org/10.1016/j.ijsolstr.2015.11.018.

[31] Yin A, Yang X, Zhang C, Zeng G, Yang Z. Three-dimensional heterogeneous fracture simulation of asphalt mixture under uniaxial tension with cohesive crack model. Constr Build Mater 2015;76:103–17. https://doi.org/10.1016/j.conbuildmat.2014.11.065.

[32] Benkemoun N, Hautefeuille M, Colliat JB, Ibrahimbegovic A. Failure of heterogeneous materials: 3D meso-scale FE models with embedded discontinuities. Int J Numer Methods Eng 2010;82:1671–88. https://doi.org/10.1002/NME.2816.

[33] Caballero A, López CM, Carol I. 3D meso-structural analysis of concrete specimens under uniaxial tension. Comput Methods Appl Mech Eng 2006;195:7182–95. https://doi.org/10.1016/j.cma.2005.05.052.

[34] Galindo-Torres SA, Pedroso DM, Williams DJ, Li L. Breaking processes in three-dimensional bonded granular materials with general shapes. Comput Phys Commun 2012;183:266–77. https://doi.org/10.1016/j.cpc.2011.10.001.

[35] Nagai K, Sato Y, Ueda T. Mesoscopic simulation of failure of mortar and concrete by 3D RBSM. Journal of Advanced Concrete Technology 2005;3:385–402. https://doi.org/10.3151/jact.3.385.

[36] Zhang J, Wang Z, Yang H, Wang Z, Shu X. 3D meso-scale modeling of reinforcement concrete with high volume fraction of randomly distributed aggregates. Constr Build Mater 2018;164:350–61. https://doi.org/10.1016/j.conbuildmat.2017.12.229.

[37] Huang Y, Yang Z, Ren W, Liu G, Zhang C. 3D meso-scale fracture modelling and validation of concrete based on in-situ X-ray Computed Tomography images using damage plasticity model. Int J Solids Struct 2015;67:340–52. https://doi.org/10.1016/j.ijsolstr.2015.05.002.

[38] Man HK, Van Mier JGM. Damage distribution and size effect in numerical concrete from lattice analyses. Cem Concr Compos 2011;33:867–80. https://doi.org/10.1016/j.cemconcomp.2011.01.008.

[39] Zhou R, Song Z, Lu Y. 3D mesoscale finite element modelling of concrete. Comput Struct 2017;192:96–113. https://doi.org/10.1016/J.COMPSTRUC.2017.07.009.

[40] Benkemoun N, Hautefeuille M, Colliat JB, Ibrahimbegovic A. Failure of heterogeneous materials: 3D meso-scale FE models with embedded discontinuities. Int J Numer Methods Eng 2010;82:1671–88. https://doi.org/10.1002/NME.2816.

[41] Ouyang H, Chen X. 3D meso-scale modeling of concrete with a local background grid method. Constr Build Mater 2020;257. https://doi.org/10.1016/j.conbuildmat.2020.119382.

[42] Zheng Z, Wei X, Tian C. Mesoscale models and uniaxial tensile numerical simulations of concrete considering material heterogeneity and spatial correlation. Constr Build Mater 2021;312. https://doi.org/10.1016/j.conbuildmat.2021.125428.

[43] Ma D, Liang Z, Liu Y, Jiang Z, Liu Z, Zhou L, et al. Mesoscale modeling of epoxy polymer concrete under tension or bending. Compos Struct 2021;256. https://doi.org/10.1016/j.compstruct.2020.113079.

[44] Wang X, Liang Z, Nie Z, Gong J. Stochastic numerical model of stone-based materials with realistic stone-inclusion features. Constr Build Mater 2019;197:830–48. https://doi.org/10.1016/j.conbuildmat.2018.10.062.

[45] Thirumalaiselvi A, Anandavalli N, Rajasankar J. Mesoscale studies on the effect of aggregate shape idealisation in concrete. Magazine of Concrete Research 2019;71:244–59. https://doi.org/10.1680/jmacr.17.00184.

[46] Alexander M, Mindess S. Aggregates in Concrete. Aggregates in Concrete 2005. https://doi.org/10.1201/9781482264647.

[47] Ren Q, Pacheco J, de Brito J. Methods for the modelling of concrete mesostructures: a critical review. Constr Build Mater 2023;408:133570. https://doi.org/10.1016/J.CONBUILDMAT.2023.133570.

[48] Zhang Z, Song X, Liu Y, Wu D, Song C. Three-dimensional mesoscale modelling of concrete composites by using random walking algorithm. Compos Sci Technol 2017;149:235–45. https://doi.org/10.1016/j.compscitech.2017.06.015.

[49] Wriggers P, Moftah SO. Mesoscale models for concrete: Homogenisation and damage behaviour. Finite Elements in Analysis and Design 2006;42:623–36. https://doi.org/10.1016/J.FINEL.2005.11.008.

[50] Gal E, Ganz A, Hadad L, Kryvoruk R. Development of a Concrete Unit Cell. Int J Multiscale Comput Eng 2008;6:499–510. https://doi.org/10.1615/INTJMULTCOMPENG.V6.I5.80.

[51] Shahbeyk S, Hosseini M, Yaghoobi M. Mesoscale finite element prediction of concrete failure 2011. https://doi.org/10.1016/j.commatsci.2011.01.044.