RAPToRS – **R**apid **A**nalysis and **P**rocessing **To**ol for **R**andom Granular **S**tructures

P.G. Bolz

Technische Universität Dresden, Department of Civil Engineering, Institute for Urban and Pavement Engineering, Georg-Schumann-Straße 7, 01187 Dresden, Saxony, Germany

ABSTRACT: The *RAPToRS* tool (Rapid Analysis and Processing Tool for Random Granular Structures) is a *Python*-based application designed for efficient generation, analysis, and processing of granular structures with specified grain parameters. It supports the serial calculation of mechanical properties for composite granular materials, such as concrete and asphalt, enabling applications such as AI-driven property prediction and statistical material analysis. *RAPToRS* leverages *Python's* parallelisation capabilities for enhanced computational performance. In test calculations, it generated and analysed 100 random models in about 64 s per model on a single processor. Integrating with *OpenSeesPy's* extensive library of 73 material models, *RAPToRS* also facilitates advanced analyses of complex material behaviours of multiphase materials with varied properties.

1 PARTICLE SHAPE GENERATION

1.1 *Shape Generation Algorithms*

The generation of randomly shaped particles and the detection of overlapping are crucial for the numerical simulation of granular structures. Among others, the following algorithms and methods are used for the generation of irregularly shaped particles: R(θ) method (Wang et al. 1999), Voronoi grain-based method (Liu et al. 2018), (Mollon & Zhao 2012), (Emig et al. 2023), digital image-based method (Shan & Lai 2019), image-based clump library method (Zheng & Hryciw 2016), (Zheng & Hryciw 2017), Fourier-based method (Lianheng et al. 2017), (X. Wang et al. 2019), (Nie et al. 2019) and the Z-R shape function (Z. Wang et al. 2019). However, the majority of these algorithms are time-consuming and computationally complex to implement, meaning that a large number of calculations can only be realised to a limited extent.

1.2 *Shape Descriptors*

Numerous studies have attempted to quantitatively analyse the geometric properties of particles or grains. These morphological characteristics are typically described in three main dimensions: form (overall shape), roundness (angularity) and surface texture (roughness) (Lianheng et al. 2017), (Wang et al. 2005), (Blott & Pye 2008).

Form, considered the primary characteristic, refers to large-scale spatial irregularities in particle shape. Common shapes include circles, ellipses, rectangles, etc. To quantify a particle's basic shape, descriptors such as elongation index (EI), aspect ratio, and flatness index are used.

The EI is determined using the width W and length L of the smallest rectangular box containing the particle as follows (Mollon & Zhao 2012):

$$EI = W/L. \tag{1}$$

Roundness, a secondary characteristic, refers to variations in particle shape at the intermediate scale, capturing the average sharpness of the particle's corners. It is not influenced by the overall shape. Shape descriptors such as the angularity index, roundness index (RI) and others are used to quantify this feature. The degree of roundness is determined and calculated as follows (Janoo 1998):

$$RI = 4\pi A/P^2 \tag{2}$$

where A is the projected area, while P is the perimeter of the particle contour.

Surface texture, a tertiary characteristic, refers to small-scale surface features relative to the particle's overall size. The roughness index and regularity index are commonly used methods to quantify surface texture. The regularity index (RE) is calculated as (Mollon & Zhao 2012):

$$RE = \log\left(\frac{P}{P-P_{conv}}\right) \tag{3}$$

where P is the perimeter and P_{conv} is the convex perimeter of the particle.

1.3 *New Approach for Particle Shape Generation*

A new and simple approach was developed using the *Python* programming language. This approach uses the *polygenerator* package, made available by Radovan Bast in 2021, which is licensed under the MIT License and can be obtained via *GitHub* (Bast 2021a) or *pip* (Bast 2021b). The code of the package or the method for generating polygons will not be discussed in detail here, but the code is available on *GitHub*.

The package provides three functions for generating random polygons. The function *random_polygon()* creates random and chaotic polygons, whereas the function *random_star_shapd_polygon()* yields random star shaped polygons as the name suggests. In this paper, the function *random_convex_polygon()* was used for creating random convex polygons with a specific number of vertices as a function argument.

In order to ensure the reproducibility of polygons using this function, the random state was initially set by the *Python* function *random.seed()* before creating the polygon. Within a for-loop from 0 to 10209 for the initial random state, 98 polygons were created per random state with a nested for-loop from 3 to 100 number of vertices. A total of 1,000,580 different polygons were thus created, the shape descriptors and other parameters were determined and the parameter sets were saved in a text file. By re-importing the text file, polygons that satisfy certain conditions can be recreated rapidly and efficiently.

Figure 1 illustrates the relationship between the minimal polygon size, the number of polygon vertices and the elongation index according to equation (1) and Figure 2 depicts the relationship between the polygon area, the number of vertices and the roundness index according to equation (2) for all 1,000,580 polygons. All polygons are created in a square box of 1 mm by 1 mm, hence the range for the minimal polygon size from almost 0 for very elongated polygons (EI ≈ 0) with a small number of vertices to 1 for polygons resembling almost perfect circles with a high number of vertices and a ratio of width to length of 1. Given that the roundness index for these polygons is equal to 1, the polygon area corresponds to the area of a circle with diameter 1 mm (0,785 mm²) as can be seen in Figure 2. Since the regularity index for convex polygons approaches infinity according to equation (3), this index is not considered further here.

2 MODEL GENERATION AND ANALYSIS

2.1 *Granular Structure Generation*

The initial step is to define the dimensions of the model in the x and y direction. Using *Python*, an algorithm was written that starts by taking a random subset with a specific number of grains (polygons) matching the user-defined criteria from the entirety of the aforementioned 1,000,580 polygons. The grains are then scaled in size and randomly arranged within the model boundaries. This process starts with the largest grains and the size of the grains is gradually reduced until all grains are placed within the model.

The algorithm checks for each grain if it overlaps with or is entirely within the boundaries of an already placed grain. If the former is the case, the grain is rotated up to 360° on the spot to check whether it fits into the current granular structure in a different orientation. If this also fails or the grain is inside another grain, a new position for the grain is randomly initiated and the algorithm starts anew.

This method enables the creation of random two-dimensional granular structures consisting of grains with a specific shape and size that can be created with high temporal and computational efficiency. Figure 3 depicts an exemplary random granular structure with dimensions 125 mm by 125 mm and 100 grains with EI between 0.45 and 0.55, a max. grain size of 16 mm and a min. grain size of 5 mm that was created using the described algorithm. The determination of the resulting particle size distribution is implemented in the algorithm, but is not considered further here.

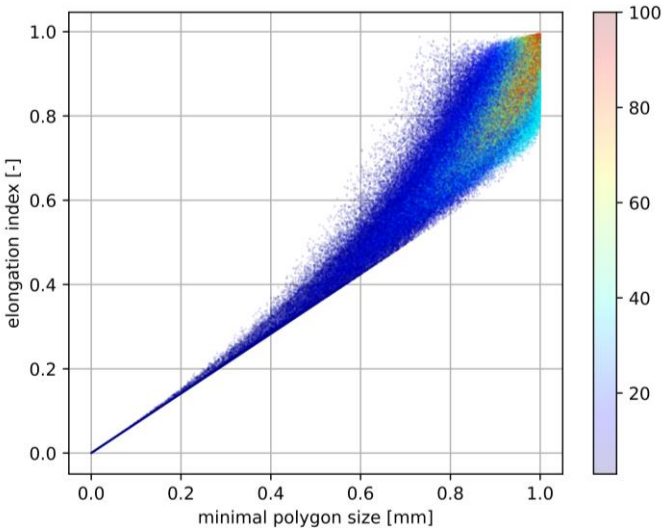

Figure 1. Minimal polygon size vs. elongation index for all 1,000,580 polygons with colour-coded number of vertices

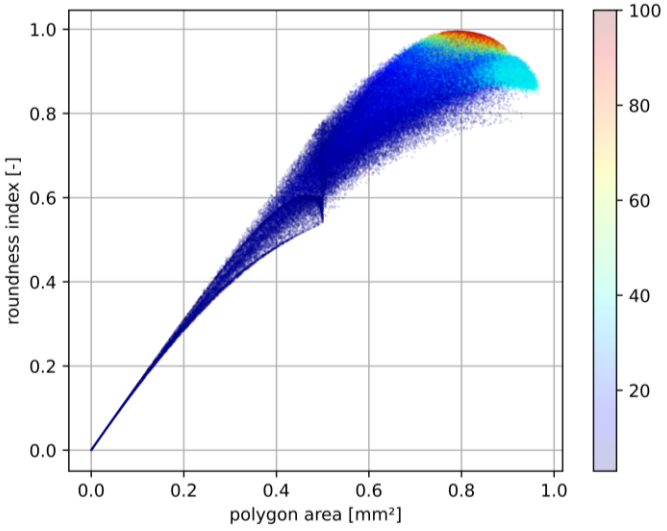

Figure 2. Polygon area vs. roundness index for all 1,000,580 polygons with colour-coded number of vertices

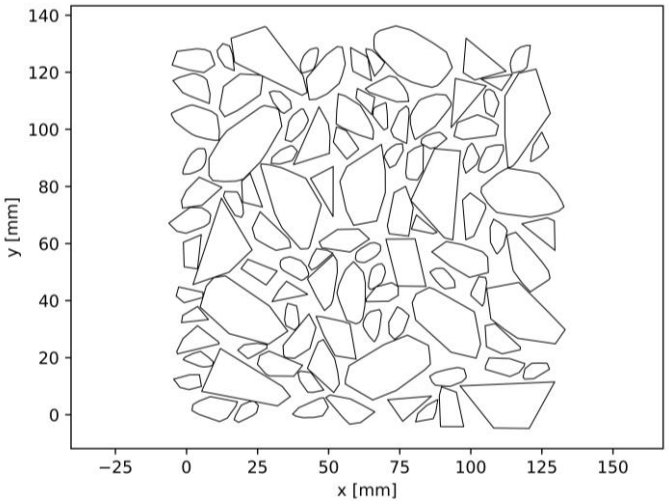

Figure 3: Random granular structure with dimensions 125 mm by 125 mm consisting of 100 grains with an elongation index between 0.45 and 0.55, a maximum grain size of 16 mm and a minimum grain size of 5 mm

2.2 *Model Generation*

The actual model for calculating the deformations and stresses of the granular structure is created using *OpenSeesPy*. This is a version of the *OpenSees* finite element software framework adapted for use in *Python*, which is maintained by the University of California at Berkeley. Further information on *OpenSees* and *OpenSeesPy* can be found in the respective documentation (OpenSees 2024), (OpenSeesPy 2024).

The granular structure model is generated by first choosing a resolution for the model and by then iterating over every centre point of the resulting discreet model and checking, whether the point lies within a grain (polygon) of the granular structure. If this is the case, the designated model parameters for the grains or aggregates are chosen. If otherwise the point lies within the mastic or cement stone, the respective model parameters are selected. A *FourNodeQuad element* which uses a bilinear isoparametric formulation is then created at the position of the centre point with model parameters accordingly using the four adjacent corner nodes. In order to accelerate this process, a KDTree is used for faster assignment of the elements to the respective material phases.

Thus, a plane stress / plane strain elastic model with thickness t = 1 mm is created for which boundary conditions can be defined. Figure 4 depicts the model created in this way for the exemplary granular structure mentioned in the last section.

Note that the grains are cut off at the specified edges of the model. Using this approach, a variety of different model geometries can be created. For example, a circular disc can be considered by checking whether the centre points of the model lie inside a circle with a defined radius. Thus, the indirect tensile test, which is widely used in pavement engineering can be modelled.

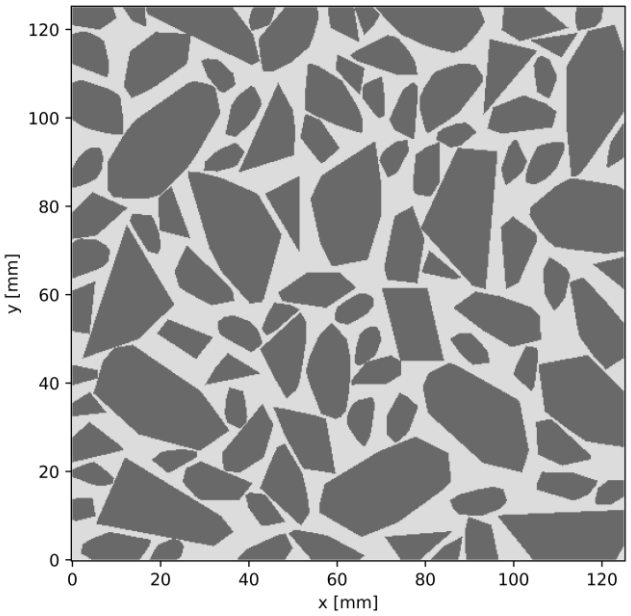

Figure 4. Generated granular model using *OpenSeesPy* with a resolution of 0.25 mm and thickness t = 1 mm; dark grey: aggregates / grains with E = 80,000 N/mm² and ν = 0.2; light grey: cement stone with E = 5,000 N/mm² and ν = 0.3

2.3 *Model Analysis and Processing*

Continuing with the previously considered example, the following boundary conditions are applied to explain the model analysis using *OpenSeesPy*. The displacement of the nodes at y = 0 is fixed in x and y direction and the displacement of the nodes at y = 125 is fixed in x direction. Additionally, a multi-point constraint is defined between the nodes at y = 125, forcing a synchronised movement in y direction. A force of 1000 N is equally distributed over all nodes at y = 125. The model is subsequently analysed using the respective *Analysis Commands* provided by *OpenSeesPy* in *Python*. Figure 5 depicts the resulting von Mises stress in the exemplary model with the aforementioned boundary conditions.

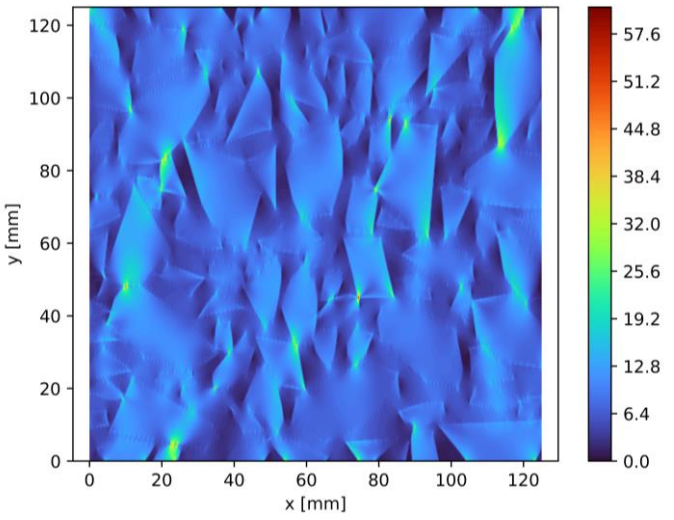

Figure 5. Von Mises stress in N/mm² of the granular model with defined boundary conditions plotted using *Opsvis*

As all results are available in *Python*, further analyses or processing of the results or the model in general can be carried out conveniently via efficient modules such as *NumPy* or *matplotlib*. For example, the visualisation presented in Figure 5 is the result of the *Opsvis* module (Opsvis 2024), which is based on *matplotlib* and was designed specifically for plotting models and results of *OpenSeesPy*.

3 CONCLUSION AND OUTLOOK

The collection of self-written algorithms and *Python* modules and packages described in this paper is referred to as the **R**apid **A**nalysis and **P**rocessing **To**ol for **R**andom Granular **S**tructures – *RAPToRS*. This tool allows for the time and computational effective generation, analysis and processing of random granular structures consisting of grains with specific parameters all within the *Python* programming language.

The *RAPToRS* tool enables the realisation of serial calculations of mechanical properties of all kinds of composite granular materials such as concrete or asphalt. This data can then be used e.g., for the learning of artificial intelligence metamodels for the prediction of said properties or for the statistical analysis of the properties of one specific material. As the tool is written entirely in *Python*, other useful modules and packages can be used, e.g. for the parallelisation of calculations.

A test calculation of 100 random models each measuring 125 mm by 125 mm with 80 grains ($0.45 < EI < 0.55$, 16 mm maximum and 5 mm minimum grain size) and a resolution of 0.5 mm using one processor (AMD Ryzen 7 PRO 4750U) took 64.17 s on average (standard deviation of 1.44 s). Computing 10,000 models with the same parameters using a moderately performant computer with 64 processors would therefore take around 167 min or 2.8 h.

The *OpenSeesPy* framework currently offers a total of 73 different material models for linear, non-linear, elastic, viscous, plastic and other complex material behaviour. This allows not only trivial calculations to be carried out, as demonstrated in this paper, but also complex relationships resulting from the granular material structure to be analysed. The presented approach for generating granular structures also allows the consideration of multiphase materials with an arbitrary amount of material phases.

It should be noted however, that the tool is not without limitations. On the one hand, only two-dimensional, plane stress / plane strain problems can be considered. Secondly, no contact conditions between the individual material components are currently taken into account. The aggregates are firmly embedded in the mastic. Further development of the tool may allow contacts to be taken into account in the future, e.g. by implementing interface elements. Additionally, only convex polygons (particle shapes) are currently considered. However, concave polygons can also be generated using the *polygenerator* package, which will be implemented in the *RAPToRS* tool in the future. Finally, the tool has not yet been validated. Validation can be realised, e.g., by comparing the granular structures generated using the tool with real structures.

REFERENCES

Wang, Z. M. et al. 1999. Mesoscopic study of concrete I: generation of random aggregate structure and finite element mesh. *Computers & structures, 70(5), 533-544*.

Liu, Q. et al. 2018. A Voronoi element based-numerical manifold method (VE-NMM) for investigating micro/macro-mechanical properties of intact rocks. *Engineering Fracture Mechanics*, *199*, 71-85.

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

Emig, J. et al. 2023. A stochastic neural network based approach for metamodelling of mechanical asphalt concrete properties. International Journal of Pavement Engineering 24.1: 2177650.

Shan, P. & Lai, X. 2019. Mesoscopic structure PFC∼ 2D model of soil rock mixture based on digital image. *Journal of Visual Communication and Image Representation*, *58*, 407-415.

Zheng, J. & Hryciw, R. D. 2016. A corner preserving algorithm for realistic DEM soil particle generation. *Granular Matter*, *18*(4), 84.

Zheng, J. & Hryciw, R. D. 2017. An image based clump library for DEM simulations. *Granular Matter, 19(2), 26*.

Lianheng, Z. et al. 2017. Reconstruction of granular railway ballast based on inverse discrete Fourier transform method. *Granular Matter 19*: 1-17.

Wang, X. et al. 2019. Stochastic numerical model of stone-based materials with realistic stone-inclusion features. *Construction and Building Materials, 197, 830-848*.

Nie, Z. et al. 2019. Investigating the effects of Fourier-based particle shape on the shear behaviors of rockfill material via DEM. *Granular Matter, 21,* 1-15.

Wang, Z. et al. 2019. A random angular bend algorithm for two-dimensional discrete modeling of granular materials. *Materials*, *12*(13), 2169.

Wang, L. et al. 2005. Unified method to quantify aggregate shape angularity and texture using Fourier analysis. *Journal of Materials in Civil Engineering 17.5*: 498-504.

Blott, S. J. & Pye, K. 2008. Particle shape: a review and new methods of characterization and classification. *Sedimentology 55.1*: 31-63.

Janoo, V. C. 1998. Quantification of shape, angularity, and surface texture of base course materials.

Bast, R (a). 2021. Polygenerator. URL: https://github.com/bast/polygenerator. Retrieved on November 5, 2024.

Bast, R (b). 2021. Polygenerator. URL: https://pypi.org/project/polygenerator/. Retrieved on November 5, 2024.

OpenSees. 2024. OpenSees Online Documentation. URL: https://opensees.berkeley.edu/index.php. Retrieved on November 5, 2024.

OpenSeesPy. 2024. OpenSeesPy Online Documentation. URL: https://openseespydoc.readthedocs.io/en/latest/. Retrieved on November 5, 2024.

Opsvis. 2024. Opsvis Online Documentation. URL: https://opsvis.readthedocs.io/en/latest/. Retrieved on November 5, 2024.