Acta Polytechnica CTU Proceedings **54**:17–22, 2025
https://doi.org/10.14311/APP.2025.54.0017

Published by the Czech Technical University in Prague

TRACTABLE DESCRIPTORS FOR DIGITAL AGGREGATE GENERATION

Kaustav Das, Jan Sýkora*, Anna Kučerová

Czech Technical University in Prague, Faculty of Civil Engineering, Department of Mechanics, Thákurova 2077/7, 160 00 Prague 6 – Dejvice, Czech Republic

* corresponding author: jan.sykora.1@cvut.cz

Abstract. This paper introduces a deep learning-based approach for generating 2D aggregate shapes using a small set of tractable and physically meaningful parameters. In contrast to existing methods that often rely on quantities without clear physical interpretations, the approach presented here leverages scale invariant moments to capture essential shape characteristics. Additionally, we introduce the idea of using scale-invariant state descriptors, such as area, to control the size of the generated shapes. The neural network is trained to generate shapes corresponding to these parameters, and its ability to learn the relationship between shape constants and state descriptors without explicit data augmentation is demonstrated. The framework thus presented provides a foundation for developing microstructure generators that offer enhanced interpretability by relying on parameters that provide meaningful insights into the description of material morphology composed of non-trivial shapes.

Keywords: Reconstruction, moment invariants, deep learning, concrete.

1. Introduction

Concrete is primarily a matrix inclusion composite consisting of a mortar-based matrix embedded with aggregate-based inclusions. Since the casting process results in the introduction of porosity defects and also leads to the formation of *interfacial transition zones* (ITZ) in the regions between the aggregates and the surrounding mortar, these are also treated as additional phases during analysis [1, 2]. The heterogeneous structure formed by the spatial arrangement of these constitutive phases is herein termed the concrete microstructure.

The role of concrete microstructure on its overall performance is well established with correlation being observed in its mechanical behaviour [3], crack formation [4], hydration mechanisms [5], thermo-mechanical coupling [6] to name a few. In particular, the size, shape and the roughness of aggregates play an important role in the overall strength of concretes [7] and the development of ITZs [8], which are accepted as the weakest phase in concretes [9].

However, the grading of aggregates, their property, and their type are also among the controllable parameters in concrete mixtures. Therefore, due to the vast number of possible combinations of parameters, development of concrete mixtures for specific applications requires substantial investment of time and resources if performed by exclusively employing experimental methods. Alternatively, numerical schemes enable an effective preliminary study of concretes and can greatly reduce experiments by facilitating the inspection of a larger set of parameters in-silico [2].

For materials such as concretes where scale separation is well defined, multi-scale homogenization schemes are used for determining the bulk response by taking into consideration material heterogeneity [10]. However, accuracy of numerical schemes depend upon the level of fidelity used for the definition of the computational domain which in this case, is linked to the distribution of aggregates.

Generation of complex computational domains or microstructures fall into the purview of *microstructure characterization and reconstruction* (MCR) techniques where some of the widely adopted methods for modeling heterogeneous material structures either include the use of very large set of parameters or include the use of specialized functions without well defined physical meaning [11]. This is also evident in the modeling of aggregates where spherical harmonics and random fields see wide adoption [12], which by themselves lack clear physical meanings. Although it is possible to obtain a limited number of physical descriptors based on polygonal fits such as bounding boxes, circumscribed circles or ellipses etc, they are rarely used for generation of aggregates but are rather used for general evaluation [12].

This paper focuses on the utilization of deep-learning (DL) tool for generation of aggregate shapes in 2D by utilizing a small set of tractable and physically meaningful parameters, which are henceforth termed as *geometric descriptors* [13] of an aggregate. In this effect, the idea of moment invaraints for image recognition is revisited which was originally proposed in [14] and improved upon in [15]. For more information on image moments, one may refer to [16]. A key highlight in these works is in the use of geometric descriptors that are shape specific and do not change under certain transformations.

This idea, as a consequence, leads us to the possibility of further categorizing geometric descriptors

into two distinct classes: first, descriptors that are shape specific and are invariant to certain types of transformations, and second, descriptors that specify the state of the shape, such as size or orientation etc. Hence, from hereon, we refer to the first type as *shape constant* (SC) and the second type as *state descriptors* (SD). As is demonstrated later, for any shape represented using a set of SCs, SD adds a level of tunability for each shape.

In the subsequent sections, we focus on moments of binary images that are invariant to both translation and scale, and treat them as shape constants. Subsequently, we train a neural network to generate shapes corresponding to these moments. Moreover, we condition the network over the area of the shapes such that the network is able to generate similar shapes of varying sizes. This is despite the fact that the data-set is not explicitly augmented to provide any relation between aggregate size and its shape constants. Thereby highlighting the ability of the model to learn the mutual independence between the shape invariants and the chosen state descriptor.

Therefore, in this work, Section 2 presents the overall methodology of obtining the shape invariants and details about the neural network architecture that is used. This is followed by the results in Section 3 and is finally followed by the conclusion derived from this work in Section 4.

2. Methodology

2.1. Basic theory of image moments

Computation of image moments relies on defining an image as an intensity distribution function $\rho(x, y)$, which in the case of single-channel binary images is given as $I(x, y)$, where, $I(x, y) = 1$ if the point (x, y) is in the region of interest or else $I(x, y) = 0$. As such, the general expression of a generic moment M_{pq} of $(p + q)^{\text{th}}$ order is given as:

$$M_{pq} = \mathcal{N} \sum_{i=0}^{p} \sum_{j=0}^{q} \mathcal{K}(x, y) I(x, y), \tag{1}$$

where, $\mathcal{N}$ is the normalizing factor and $\mathcal{K}(x, y)$ is the moment kernel.

For raw-geometric moments, denoted using m_{pq}, $\mathcal{N} = 1$ and $\mathcal{K}(x, y) = x^p y^q$. The raw moment m_{00} indicates the area of the region of interest and $(\bar{x}, \bar{y})$ gives its centroid such that $\bar{x} = m_{10}/m_{00}$, $\bar{y} = m_{01}/m_{00}$.

Since this work particularly focuses on translation and scale invariant moments of aggregates, for any translation invariant moment, also known as central moment, and denoted using μ_{pq}, in which case, $\mathcal{N} = 1$ and $\mathcal{K}(x, y) = (x - \bar{x})^p (y - \bar{y})^q$, the scale and translation invariant moment, also known as normalized-central moments are given by:

$$\eta_{pq} = \frac{1}{\mu_{00}^{(1+\frac{p+q}{2})}} \mu_{pq}. \tag{2}$$

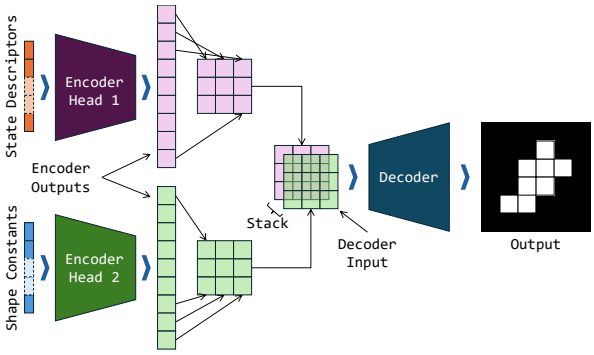

Figure 1. General architecture of the aggregate generation network.

Consequently, the final expression for scale and translation invariant moments i.e. normalized-central moments is given as:

$$\eta_{pq} = \frac{1}{\mu_{00}^{(1+\frac{p+q}{2})}} \sum_{i=0}^{p} \sum_{j=0}^{q} (x - \bar{x})^p (y - \bar{y})^q I(x, y). \tag{3}$$

In the subsequent work, moments derived using Equation 3 is used for quantification of aggregate shapes, while m_{00} is used as the state descriptor for the area. It is to be noted that $\eta_{00} = 1$ and $\eta_{01} = \eta_{10} = 0$ and are therefore not used. Therefore, the sets of image moments used are given as $\{m_{00}\}$, which is the state descriptor (SD) and $\{\eta_{20}, \eta_{11}, \cdots, \eta_{70}, \eta_{61}, \eta_{52}, \cdots, \eta_{07}\}$, which is the set of shape constants (SCs).

2.2. Neural network for aggregate generation

Given a set of SD and SC, the objective of this work is to demonstrate the ability of a deep neural network to learn and generate the shape that best fits the description given using and SD and SC. To this effect, a multi-headed autoencoder is implemented that makes use of two encoder-heads to encode the SC and SD respectively. The outputs of each encoder head are combined and used as input for a convolution-based decoder section. The overall architecture of the model is presented in Figure 1.

Each of the encoder heads is composed of a series of fully-connceted layers, each of which apply an affine transform to the incoming vector data. For any incoming vector $\boldsymbol{u}$, such an affine transform is given by $\boldsymbol{v} = \boldsymbol{u}\mathbf{A}^T + \boldsymbol{b}$, where $\boldsymbol{v}$ is the output vector and $\boldsymbol{b}$ is a vector of constants, also known as bias. Each of the fully-connected layers is followed by a Rectified Linear Unit (ReLU), which imparts non-linearity to the model. ReLU is defined as the non-negative part of any argument and as such, $\text{ReLU}(x) = x$ if $x \geq 0$ and 0 otherwise. The transformation matrix $\mathbf{A}$ and the bias vector $\boldsymbol{b}$ is optimized through training. Details about the Encoder Head 1 is given in Table 1 and Encoder Head 2 is given in Table 2. Both the encoders are designed to have output vectors of

Layer	In/Out-features	Activation func.
Fully Connected	1/1 024	ReLU
Fully Connected	1 024/1 024	ReLU

TABLE 1. Architecture of Encoder Head 1.

Layer	In/Out-features	Activation func.
Fully Connected	33/2 048	ReLU
Fully Connected	2 048/2 048	ReLU
Fully Connected	2 048/1 024	ReLU
Fully Connected	1 024/1 024	ReLU

TABLE 2. Architecture of Encoder Head 2.

equal size. Each of the output vectors are re-shaped into a matrix and concatenated to obtain a tensor of shape [2, 32, 32], which forms the input for the decoder. The first dimension here is termed as the feature dimension and the remaining dimensions are the y and x directions in the Cartesian coordinate system respectively.

The decoder comprises of alternating transposed-convolutions and convolution layers [17]. Transposed 2D-convolution layers with a stride of 2 and kernels K of size of (4,4) is used to up-sample the images to twice the spatial dimension of the input, while the convolution layers are designed to preserve the size of the output. The depth of the decoder is adjusted such that the input to the decoder is 32 pixels in both dimensions and the output is 512 pixels in both dimensions with a feature dimension of size 1, which in this case necessitates the use of four transposed convolution layers. Details about the decoder is presented in Table 3.

Finally, the resulting model is trained to obtain a set of parameters θ. Inferencing using this model is subsequently performed by using this fixed set of parameters θ such that given a pair of Shape Constants and State Descriptors, the model is able to generate an approximate 2D aggregate indicated by these quantities. The overall workflow for training the model is presented in Figure 2 and the training details is provided in Section 2.4.

2.3. AGGREGATE DATASET

The image dataset consisted of binary images, each containing a single aggregate. Aggregates were represented with a pixel value of 1, while 0 indicated the background. The image dimensions were adjusted to the smallest power of 2 that could accommodate the largest aggregate, resulting in each image measuring 512×512 pixels. Each 1 mm of spatial dimension corresponded to 10.24 pixels. Examples of training samples are shown in Figure 3.

Training data comprised (X, Y) pairs, where X represented the input data and Y the target. Each X consisted of state descriptors (SD) and shape constants (SC). SD corresponded to the area of the aggregates,

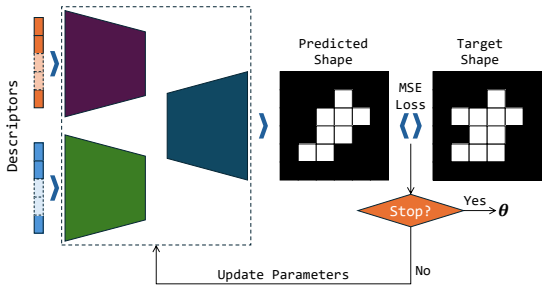

FIGURE 2. Training workflow based on supervised learning scheme.

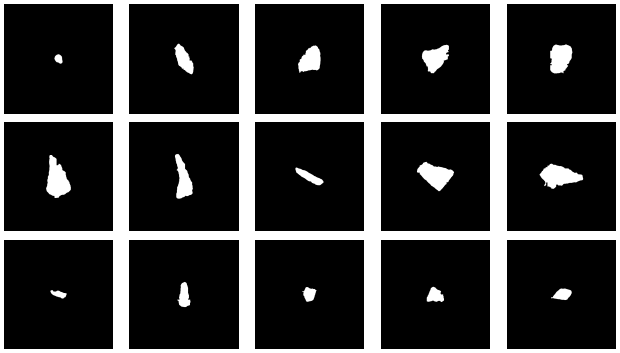

FIGURE 3. Example of training dataset containing 8 500 samples.

represented by m_{00}, and SC comprised normalized central moments, $\eta_{20}, \eta_{11}, \cdots$, up to the order $(p+q) = 7$. The target Y was the binary image of the aggregate from which X was derived.

A total of 8 500 labeled (X, Y) pairs were used for training, while a separate subset of samples was reserved for testing and excluded from training. Additionally, no data augmentation was applied to supplement the original dataset.

2.4. TRAINING

The workflow was implemented using the *PyTorch* framework [18]. The model was trained with the Adam optimizer [19] at a learning rate of 0.001, with cosine annealing applied for learning rate scheduling over 96 epochs. The mean squared error was calculated between the model's output and the target image as the loss function. Training was conducted using minibatches of 20 samples. Hyperparameters, including the learning rate, batch size, and optimizer settings, were experimentally determined to achieve optimal performance. The entire training process took approximately 2 hours on a workstation equipped with an *Nvidia RTX 4090* GPU and an *AMD Ryzen9 7950X* CPU. Inferencing with the trained model typically required about 1 minute for approximately 440 samples.

3. RESULTS

For the results presented in Figure 4, the neural network was applied to reconstruct the aggregates us-

Layer	In/Out-channels	Kernel	Stride	Padding	Activation function
TransposedConv2D	2/512	(4×4)	2	1	ReLU
Conv2D	512/512	(3×3)	1	1	ReLU
TransposedConv2D	512/256	(4×4)	2	1	ReLU
Conv2D	256/256	(3×3)	1	1	ReLU
TransposedConv2D	256/128	(4×4)	2	1	ReLU
Conv2D	128/128	(3×3)	1	1	ReLU
TransposedConv2D	128/64	(4×4)	2	1	ReLU
Conv2D	64/64	(3×3)	1	1	ReLU
Conv2D	64/64	(1×1)	1	0	ReLU

Table 3. Neural network architecture of Decoder with corresponding convolutional layers.

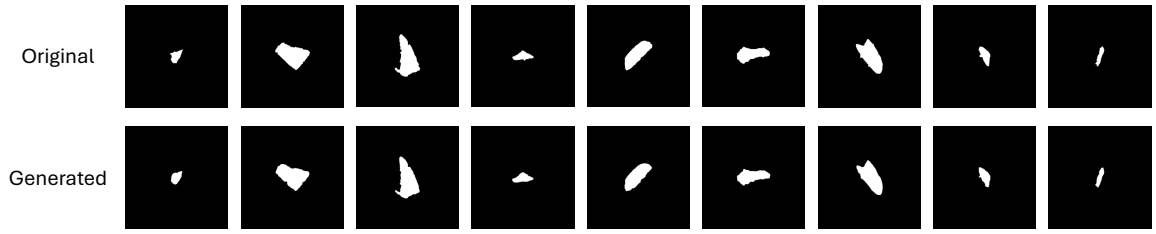

Figure 4. Reconstruction of original aggregates.

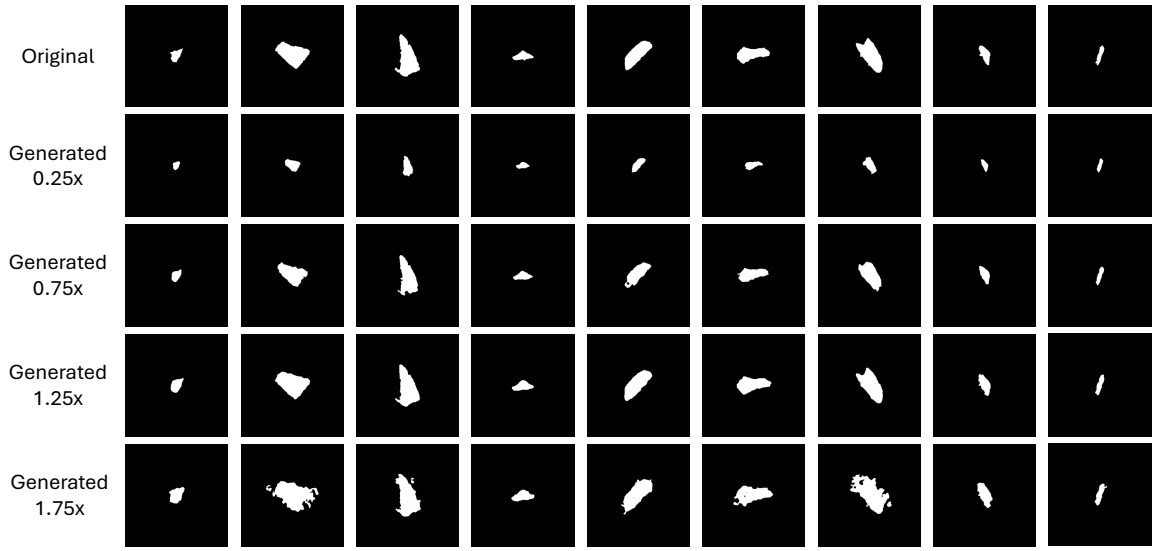

Figure 5. Microstructure generator of aggregates based on the scaled area of the original structure.

ing geometric descriptors obtained from the original images. It is evident that the model can successfully reconstruct the aggregate shapes using only the 34 quantities.

Next, to test whether the model could recognize the scale-invariant property of the normalized central moments, the model was applied to the same set of descriptors where only the area, i.e. m_{00} was scaled by a factor ranging from 0.25 to 2.0 in increments of 0.25. The results are presented in Figure 5. It is clear that the model can recognize the invariance between aggregate size and its moments. Notably, the dataset was not augmented to account for the scaling invariance of the moments, meaning this property was implicitly learned by the model.

The final results presented in Figures 4 and 5 were generated by applying the neural network to a set of descriptors that were not previously used for training. Due to the requirement of backpropagation during training, the output of the neural network is a grayscale image, with high pixel values indicating aggregate and low pixel values indicating their absence. To obtain the final binarized result, the model outputs were thresholded using Otsu's method, which is readily implemented in the *Scikit-Image* library [20].

The quality of the generated aggregates was assessed by comparing the moments of the original aggregates to those of the generated aggregates. These results are shown in Figure 6 for the various scaling factors that were used. The scaling error is presented in Figure 7,

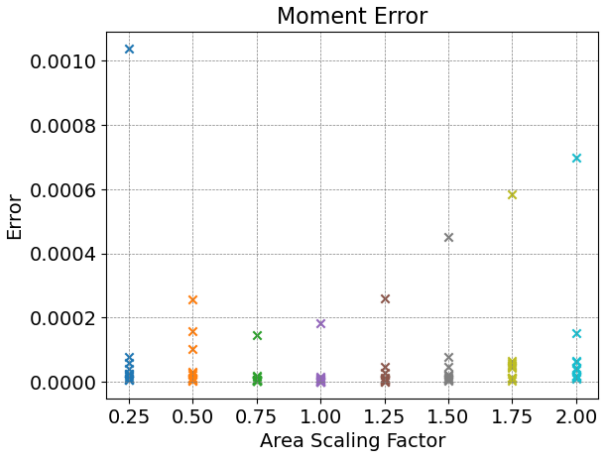

FIGURE 6. Model performance for the various scaling factors evaluated as a mean squared error (MSE) of normalized central moments.

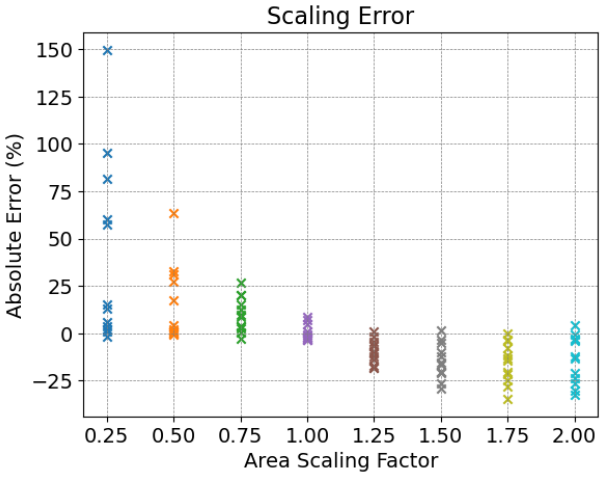

FIGURE 7. Scaling error analysis showing the model's tendency to generate larger aggregates for scaling factors less than 1 and vice versa.

where the area of the generated aggregates is compared to the correct area upon scaling. It is evident that the model tends to generate larger aggregates for scaling factors < 1, while for scaling factors > 1, it produces aggregates with smaller-than-correct areas. This may be attributed to the lack of explicit training on a dataset augmented for scale invariance of the moments.

The final study investigates how different area scaling factors affect the overall pixel wise accuracy. The results depicted in Figure 8 represent the values of the mean squared error, measuring the deviation of overlap between the predicted and original aggregates. The errors have been adjusted for scaling by dividing the MSE-error with the square of the scaling factor.

Overall, the model demonstrates good fit between the original aggregates and the predicted aggregates for a scaling factor of 1 with considerable deviation from the ideal values for other scaling factors. However, such deviation is observed to be sample specific

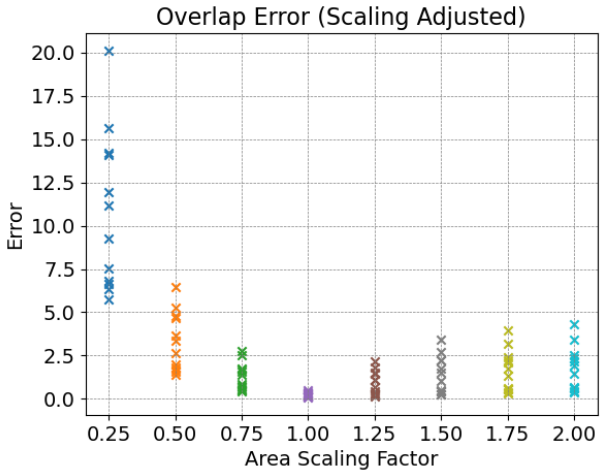

FIGURE 8. Error assessment based on MSE exploring the overlap differences between predicted and original aggregates.

with the deviation being spread over a wide range of values, with the model performing well in some cases. Nevertheless, as has already been highlighted, such a result is expected since the model was not explicitly trained to factor for scaling.

4. CONCLUSION

This paper presents a method for generating aggregate shapes in 2D using a deep-learning tool and a small set of tractable and physically meaningful parameters derived from moment invariant analysis. This approach acts as the initial step in developing a microstructure generator of heterogeneous materials that relies solely on the inputs which have clear and interpretable meanings describing material morphology. The method demonstrates the ability of a neural-network model to recognize scale variations in input prompts and adjust output sizes accordingly, despite the absence of explicit training for scale invariance. Explicit training for this aspect is expected to enhance the results further.

This method is designed with the objective of being integrating with aggregate packing schemes for microstructure generation of concretes. While the current results are focused on 2D structures, the concept can be easily extended to 3D problems, offering broader applicability in the study and design of concrete microstructures.

ACKNOWLEDGEMENTS

The authors are thankful for financial support from the Czech Science Foundation, project No. 22-35755K (KD, AK) and the Student Grant Competition of CTU, project No. SGS23/ 152/OHK1/3T/11 (JS).

REFERENCES

[1] H. C. Sleiman, M. H. Moreira, A. Tengattini, S. Dal Pont. From tomographic imaging to numerical simulations: an open-source workflow for true

morphology mesoscale FE meshes. *RILEM Technical Letters* **8**:158–164, 2023. https://doi.org/10.21809/rilemtechlett.2023.184

[2] Q. Ren, J. Pacheco, J. de Brito. Methods for the modelling of concrete mesostructures: a critical review. *Construction and Building Materials* **408**:133570, 2023. https://doi.org/10.1016/j.conbuildmat.2023.133570

[3] Z. Kammouna, M. Briffaut, Y. Malecot. Mesoscopic simulations of concrete strains incompatibilities under high creep stress level and consequences on the mechanical properties. *European Journal of Environmental and Civil Engineering* **23**(7):879–893, 2019. https://doi.org/10.1080/19648189.2017.1320235

[4] L. Li, Y. Jin, Y. Jia, et al. Influence of inclusion rigidity on shrinkage induced micro-cracking of cementitious materials. *Cement and Concrete Composites* **114**:103773, 2020. https://doi.org/10.1016/j.cemconcomp.2020.103773

[5] Z. Zheng, X. Wei. Mesoscopic models and numerical simulations of the temperature field and hydration degree in early-age concrete. *Construction and Building Materials* **266**:121001, 2021. https://doi.org/10.1016/j.conbuildmat.2020.121001

[6] T.-D. Nguyen, D.-T. Pham, M.-N. Vu. Thermo-mechanically-induced thermal conductivity change and its effect on the behaviour of concrete. *Construction and Building Materials* **198**:98–105, 2019. https://doi.org/10.1016/j.conbuildmat.2018.11.146

[7] A. Thirumalaiselvi, N. Anandavalli, J. Rajasankar. Mesoscale studies on the effect of aggregate shape idealisation in concrete. *Magazine of Concrete Research* **71**(5):244–259, 2019. https://doi.org/10.1680/jmacr.17.00184

[8] K. L. Scrivener, A. K. Crumbie, P. Laugesen. The interfacial transition zone (ITZ) between cement paste and aggregate in concrete. *Interface science* **12**:411–421, 2004. https://doi.org/10.1023/B:INTS.0000042339.92990.4c

[9] M. Nitka, J. Tejchman. Meso-mechanical modelling of damage in concrete using discrete element method with porous ITZs of defined width around aggregates. *Engineering Fracture Mechanics* **231**:107029, 2020. https://doi.org/10.1016/j.engfracmech.2020.107029

[10] K. Matouš, M. G. Geers, V. G. Kouznetsova, A. Gillman. A review of predictive nonlinear theories for multiscale modeling of heterogeneous materials. *Journal of Computational Physics* **330**:192–220, 2017. https://doi.org/10.1016/j.jcp.2016.10.070

[11] R. Bostanabad, Y. Zhang, X. Li, et al. Computational microstructure characterization and reconstruction: Review of the state-of-the-art techniques. *Progress in Materials Science* **95**:1–41, 2018. https://doi.org/10.1016/j.pmatsci.2018.01.005

[12] F.-G. Guo, H. Zhang, Z.-J. Yang, et al. A spherical harmonic-random field coupled method for efficient reconstruction of CT-image based 3D aggregates with controllable multiscale morphology. *Computer Methods in Applied Mechanics and Engineering* **406**:115901, 2023. https://doi.org/10.1016/j.cma.2023.115901

[13] H. Xu, R. Liu, A. Choudhary, W. Chen. A machine learning-based design representation method for designing heterogeneous microstructures. *Journal of Mechanical Design* **137**(5):051403, 2015. https://doi.org/10.1115/1.4029768

[14] M.-K. Hu. Visual pattern recognition by moment invariants. *IRE transactions on information theory* **8**(2):179–187, 1962. https://doi.org/10.1109/TIT.1962.1057692

[15] J. Flusser, T. Suk. Pattern recognition by affine moment invariants. *Pattern recognition* **26**(1):167–174, 1993. https://doi.org/10.1016/0031-3203(93)90098-H

[16] G. A. Papakostas. Over 50 years of image moments and moment invariants. *Gate to Computer Science & Research* **1**, 2014. https://doi.org/10.15579/gcsr.vol1.ch1

[17] M. D. Zeiler, R. Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pp. 818–833. Springer, 2014. https://doi.org/10.1007/978-3-319-10590-1_53

[18] A. Paszke, S. Gross, F. Massa, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, vol. 32. 2019.

[19] D. P. Kingma, J. Ba. Adam: A method for stochastic optimization, 2014. https://doi.org/10.48550/arXiv.1412.6980

[20] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, et al. scikit-image: image processing in Python. *PeerJ* **2**:e453, 2014. https://doi.org/10.7717/peerj.453